

SONDERHEFT 3/1986 Lit. 12000/hfl. 18,-/dkr. 68,- ÖS 100,-/Sfr. 14,- DM 14,-

Das Sonderheft zum C16, C116, VC 20 und Plus 4





Viele tolle Listings

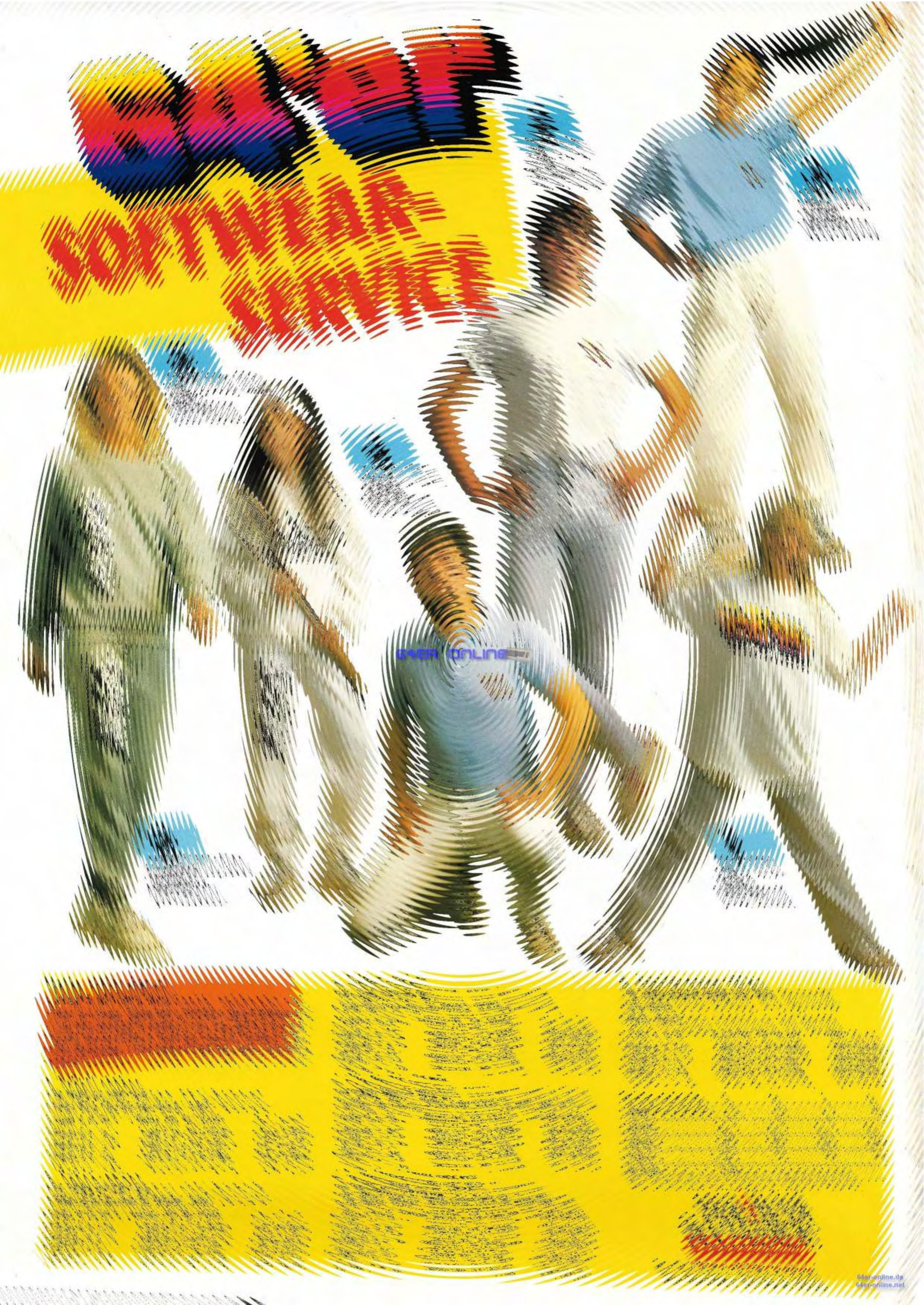
★ Super Spiele
★ Grafikerweiterungen
★ Nützliche Tips & Tricks



C16, C116 Plus 4 VC 20

★ Kurs: So programmiert man sie
★ Tips für Einsteiger
★ So arbeitet man mit der Datasette
★ Passende Drucker unter 500 Mark
★ Test: Speichererweiterungen
★ Programmierhilfen für Grafik und Spiele

Alle Programme auch auf Kassette und Diskette erhältlich



C 16/C 116 im Aufwind



Man sollte es nicht für möglich halten – und doch stimmt es: Der Commodore C 16/C 116 erlebt eine Wiederbelebungsphase sondergleichen. Schon vor seinem Erscheinen berichteten wir in der ersten 64'er-Ausgabe (4/84) vom neuen Plus 4 (damals wurde noch vom 264/364 gesprochen) und vom »Gummicomputer« C 116. Beiden Computern wurde nicht viel Erfolg prophezeit – zu Recht. Sie waren damals allesamt zu teuer. Doch zur Zeit sieht es etwas anders aus. Commodore hat die Preise purzeln lassen. Man kann durchaus von einem Ausverkauf erster Klasse sprechen. Bedenkt man, was man zur Zeit für runde 140 Mark bekommt, dann ist es schon verlockend zuzugreifen. Für so wenig Geld hat es noch nie so viel Computerleistung gegeben. Der mitgelieferte Basic-Lernkurs macht das Angebot auch für Computerscheue interessant. Und dementsprechend ist die Reaktion. Innerhalb kürzester Zeit wurden insgesamt mehr als 60000 C 16/C 116 abgesetzt.

Doch dabei soll's nicht bleiben: Commodore wird nochmal 60000 C 16/116 auf den Markt bringen, wahrscheinlich in Form eines Basic-Kurses mit Computer. Wir wissen, daß es für diese Computer relativ wenig Informationen, insbesondere Grundlagen und Programme gibt. Aus diesem Grund entstand dieses Sonderheft.

Aus einem ganz anderen Grund wollen wir die VC 20-Besitzer nicht vergessen. Dieser Oldtimer wird nicht mehr hergestellt. Jedoch existieren noch sehr viele davon. Viele ehemaligen VC 20-Besitzer haben ihn weiterverkauft oder an Freunde und Bekannte verschenkt, weil sie zum Beispiel auf den C 64 umgestiegen sind. Deshalb haben wir den VC 20 mit in dieses Sonderheft aufgenommen.

Zwar sind der VC 20 und der C 16/C 116 relativ ungleiche Brüder, doch zeigt zum Beispiel der Artikel »VC 20/C 16 durchschaut«, daß sie nicht ganz verschieden sind. Wenn in den Basic-Programmen des VC 20 (und des C 64) keine SYS-, POKE- und USR-Befehle stehen, kann jeder C 16/C 116-Besitzer diese Programme ebenfalls benutzen. Was genau geht und was nicht, lesen Sie im Artikel »C 64-Programme für C 16 und VC 20«. Wenn Sie also in unserem 64'er-Stammheft interessante Programme für den C 64 finden, ist es in vielen Fällen (wenn auch nicht immer) möglich, diese an Ihren Computer anzupassen.

Für VC 20-Besitzer haben wir noch einen Leckerbissen bereit: ein Programm, das Maschinenroutinen für den C 64 an den VC 20 anpaßt! Das gelingt in sehr vielen Fällen.

Da für den VC 20 schon sehr viel geschrieben wurde, liegt der Schwerpunkt auf den Spielelistings. Es ist fantastisch, was die Programm-Autoren aus diesem kleinen Computer noch herausgeholt haben!

Was vorhandene Literatur angeht, ist es um den C 16/C 116 schon wesentlich schlechter bestellt. Bücher sind so gut wie keine geschrieben worden, und auch Commodore selbst gab dem Computer nur wenig Lektüre mit auf den Weg. Deshalb finden C 16/C 116-Besitzer in diesem Sonderheft eine Menge Grundlagenwissen über Ihren Computer. Dazu gehört die Hardware, also die »Innereien« des C 16, die interne Software (Basic und Betriebssystem) und natürlich die Peripherie, also die Geräte, die man anschließen kann, wie Drucker, Floppy und vor allem die Datasette.

Wir zeigen Ihnen, wie man mit der Datasette umgeht, wie Dateien angelegt werden und wie man Fehler verhindert oder findet. Wenn beim Laden von Programmen Fehler auftauchen, kann es an einer falsch justierten Datasette liegen. Wir sagen Ihnen, was dann zu tun ist.

Natürlich fehlen auch Listings nicht. Es gibt viele hervorragende Programme zum Spielen und jede Menge Tips & Tricks-Listings.

Insgesamt glauben wir, daß hier jeder auf seine Kosten kommt, sowohl der VC 20- als auch der C 16/C 116-Besitzer, sowohl der Spiele-Fan als auch der Programmierer. Wenn Sie dann noch Fragen haben oder wenn Sie neue Tips oder Programme loswerden wollen, dann schreiben Sie uns doch einfach.

(Georg Klinge)

Programmservice

Wer keine Zeit oder Lust hat, alle Programme selbst in mühevoller Kleinarbeit abzuschreiben, kann wieder auf den bewährten Disketten- und Kassettenservice zugreifen. Alle Programme, die mit dem Disketten- oder Kassettensymbol im Inhaltsverzeichnis gekennzeichnet sind, gibt's auf Kassette und/oder Diskette.
Auf der Diskette sind alle gekennzeichneten Programme enthalten.

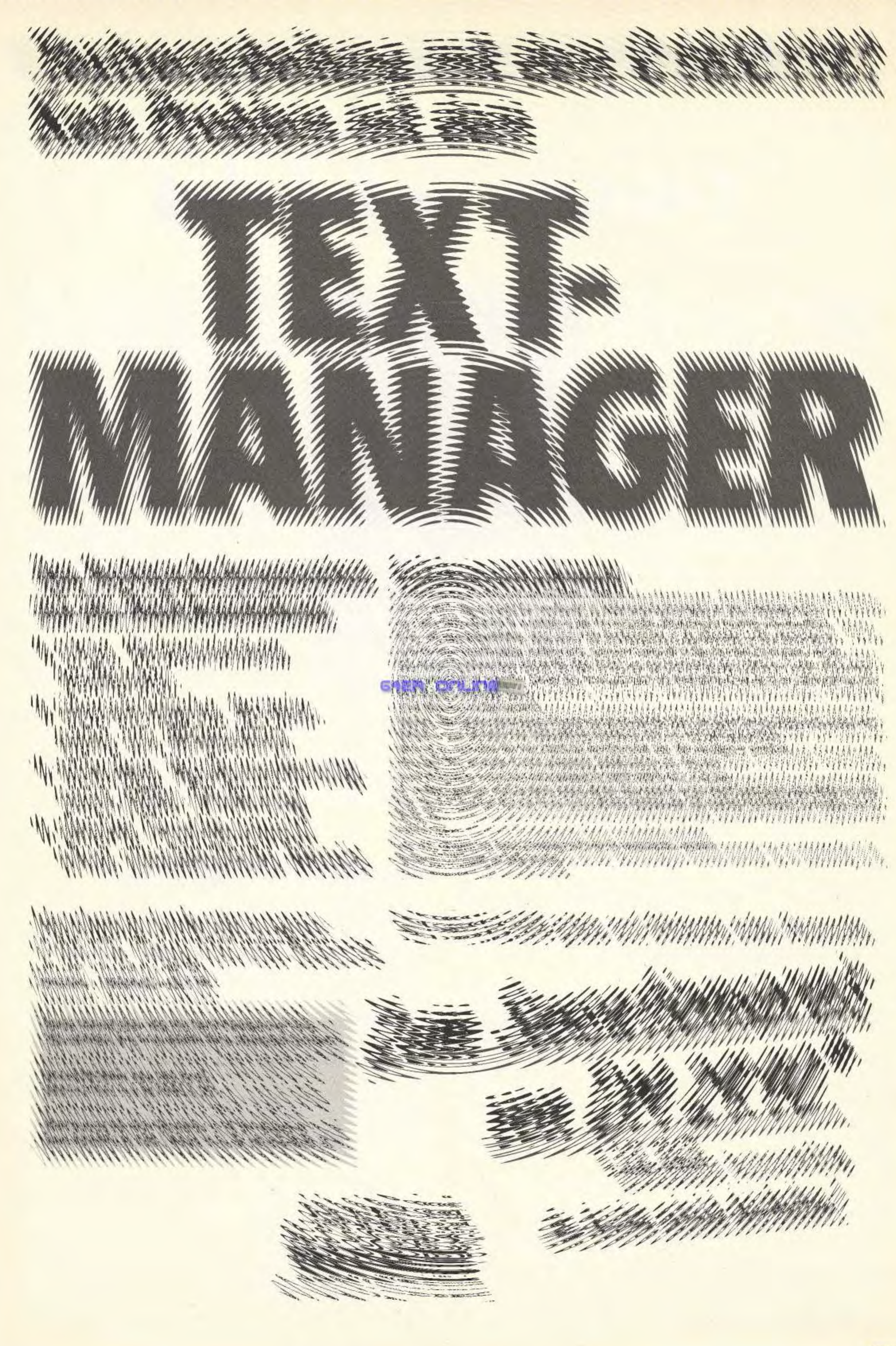
Es gibt zwei Kassetten: eine für den C 16/116 mit allen C 16/116 Programmen und eine zweite mit allen VC 20-Programmen. Es gibt einige Programme, die zwar der Vollständigkeit halber auf den Kassetten gespeichert sind, jedoch nur mit einer Diskettenstation lauffähig sind. Bitte lesen Sie deshalb die Anleitungen zu den Programmen genau durch.

Bestell-Nr. L6 86 S3 CD
(für VC 20 und C 16/116) 1 Diskette — **29,90 Mark***

Bestell-Nr. L6 86 S3 KV
(nur VC 20) 1 Kassette — **19,90 Mark***

Bestell-Nr. L6 86 S3 KC
(nur C 16/116) 1 Kassette — **19,90 Mark***

*inkl. MwSt.


TEXT-
MANAGER

INHALT

Vorwort

Grundlagen

Hardware

Bücher

Grafik

Eintipphilfe

Anwendungen

Spiele

Spieleführer (C16)

Tips und Tricks

Daten verwalten mit der Datasette

Mit der Datasette kann man nicht nur Programme laden und speichern, sondern auch Daten verwalten. Wir zeigen Ihnen, wie das gemacht wird und was dabei beachtet werden muß.

Im folgenden Artikel soll dargestellt werden, welche Möglichkeiten zur Datenspeicherung die Datasette außer dem Speichern und Laden von Programmen bietet.

Die Arbeit mit Programmen ist unproblematisch und muß daher wohl nicht näher beschrieben werden. Sie alle wissen, daß zum Speichern eines Programms der Befehl »SAVE« und zum Laden der Befehl »LOAD« zur Verfügung steht, daß der Benutzer die Möglichkeit hat, ein Programm unter einem bestimmten Namen zu speichern (SAVE»TESTPROG«), und mit dem Befehl »VERIFY« überprüfen kann, ob das Speichern fehlerfrei gelang.

Probleme können vor allem bei der Verwendung von Kassetten großer Länge auftreten. Ich persönlich verwende seit langem sogenannte »Datenkassetten« mit einer Länge von 10 Minuten pro Seite, die mir noch nie Ärger bereitet haben. Sollten Sie des öfteren einen »LOAD ERROR« oder einen »VERIFY ERROR« auf dem Bildschirm erhalten, ist wahrscheinlich entweder der Tonkopf Ihrer Datasette nicht einwandfrei justiert oder aber verschmutzt.

Ebenso wie bei jedem Kassettenrecorder sollte zuerst eine Tonkopfreinigung mit einer Reinigungskassette oder einem Reinigungsspray versucht werden. Wenn sich auch nach der Reinigung Probleme mit dem Laden von Programmen ergeben, ist der Tonkopf meist dejustiert und ein Gang zur Fachwerkstatt fällig, die über die zur einwandfreien Justierung nötigen Meßgeräte verfügt.

Die Datasette wird meistens zum Laden und Speichern von Programmen verwendet. Ihre Möglichkeiten sind damit jedoch bei weitem nicht ausgeschöpft. Mit diesem Gerät lassen sich nämlich Daten beliebiger Art verwalten, das heißt außer Programmen auch Termine, Benzinkosten, Adreß- oder Schallplattendateien und vieles mehr.

Die Verwaltung solcher Daten ist leider weitaus komplizierter als das Laden und Speichern eines normalen Programms. Im Handbuch des C 16/C 116 finden Sie dies nur sehr unzureichend beschrieben.

Grundsätzlich lassen sich mit der Datasette nur sogenannte »sequentielle« Dateien verwalten, im Gegensatz zur Floppy-Disk, die auch »Direktzugriffsdateien« verwalten kann. Wie der Name bereits sagt, kann auf alle Daten einer Direktzugriffsdatei unmittelbar zugegriffen werden. Wenn zum Beispiel die Telefonnummer des Herrn »Müller« gesucht wird, greift man auf jene Stelle in der gesamten Datei zu, an der sich der Datensatz »Müller« befindet. Der komplette Datensatz, der zum Beispiel aus den Teilen »Name: Müller«, »Vorname: Gerd«, »Adresse: München« und »Telefonnr.: 1234/56789« bestehen könnte, wird eingelesen und auf dem Bildschirm ausgegeben.

Ein solcher gezielter Zugriff auf bestimmte Teile einer Datei ist mit der Datasette leider nicht möglich. Um auf gesuchte Daten zuzugreifen, muß die gesamte Datei ab dem Dateianfang gelesen werden, bis die Bandposition erreicht wird, an der die gesuchten Daten abgespeichert wurden.

Zur Verdeutlichung des Unterschieds zwischen direktem und sequentiellem Zugriff bietet sich der Vergleich von Plattenspieler und Kassettenrecorder an: Um ein bestimmtes Lied zu hören, kann der Tonarm des Plattenspielers direkt zu jener Stelle der Schallplatte bewegt werden, an der das Stück beginnt. Beim Kassettenrecorder ist dieser direkte Zugriff nicht möglich. Das Band muß vor- oder zurückgespult werden, bis das jeweilige Lied gefunden wurde, die einzelnen Stücke werden »sequentiell« (der Reihe nach) abgesucht.

Halten wir somit fest: Mit einer Datasette als externem Speicher können nur sequentielle Dateien verwaltet werden, das heißt, Daten können nur der Reihe nach auf das Band geschrieben oder gelesen werden.

Nehmen wir nun an, Sie haben ein Programm zur Verwaltung der Adressen Ihrer Freunde, Bekannten, Verwandten etc. geschrieben, alle Adressen bereits eingetippt und gespeichert. Einen Monat später wollen Sie nun drei dieser Bekannten anschreiben, um sie zu Ihrer Geburtagsfeier einzuladen.

Da Sie sich nicht mehr exakt an die Adressen dieser Bekannten erinnern können, wollen Sie in der Adreßdatei nachschauen. Sie legen die Kassette ein, auf der sich die Datei befindet, spulen sie zum Dateianfang zurück und suchen zum Beispiel nach »Andreas«. Der erste Bekannte wird nun gesucht. Wenn er gefunden wird, gibt Ihr Programm die komplette Adresse auf dem Bildschirm aus, und Sie können den ersten Brief adressieren.

Nun spulen Sie die Kassette wieder zum Dateianfang zurück, suchen nach dem zweiten Bekannten und notieren sich auch dessen Adresse, sobald sie auf dem Bildschirm erscheint. Die Kassette wird nun ein letztes Mal zurückgespult, und die Suche nach der dritten Adresse kann beginnen.

Wie Sie sehen, ist diese Art des Umgangs mit Ihrer Adreßdatei außerordentlich unkomfortabel. Jedesmal wenn bestimmte Daten gesucht werden sollen, müssen Sie die Kassette bis zum Anfang der Datei zurückspulen. Da die Datasette nicht gerade der schnellste Massenspeicher ist, kann die Suche geraume Zeit in Anspruch nehmen, je nachdem, wie umfangreich Ihre Datei ist.

In der Praxis wählt man daher eine andere Methode des Umgangs mit sequentiellen Dateien: Zu Beginn der Arbeit lädt man die komplette Datei vom Band in den Computerspeicher, das heißt, man liest jeden einzelnen Datensatz ein. Wird eine bestimmte Adresse gesucht, erfolgt die Suche nicht mehr auf der Kassette, sondern im Computerspeicher, der ja nun alle Adressen enthält. Diese Suche im Speicher des C 16 verläuft erheblich schneller als die Suche auf der Kassette. Der Nachteil dieser Methode besteht in der Beschränkung der maximalen Dateigröße. Da alle Daten in den Computer geladen werden, darf die Datei höchstens ebenso groß sein wie der verfügbare freie Speicherplatz. Beim C 16 stehen 12 KByte an freiem Speicher zur Verfügung, von dem jedoch der Platz abgezogen werden muß, den das Dateiverwaltungsprogramm selbst belegt. Da jedes Programm Variablen enthält, die ebenfalls Platz im Speicher benötigen, wird der für die Datei verfügbare freie Computerspeicher noch weiter eingeschränkt.

Ein Beispiel: Nehmen wir an, das Programm und die von ihm benötigten Variablen belegen 6 KByte. Für die Datei stehen somit noch 12-6=6 KByte zur Verfügung. Da ein Kilobyte 1024 Zeichen entspricht, darf die Datei bis zu 6*1024=6144 Zeichen enthalten. Wenn eine Adresse, die aus den Teilen »Name«, »Vorname«, »Straße«, »Wohnort« und »Telefonnr.« besteht, im Durchschnitt zirka 80 Zeichen enthält, kann die Datei etwa 6144/80=76,8 Datensätze enthalten. Wir sehen an diesem Beispiel, daß uns der C 16 in Verbin-

dung mit der Datasette die Verwaltung kleinerer Adreßdateien erlaubt, jedoch ungeeignet ist für den kaufmännischen Einsatz, zum Beispiel die Verwaltung eines Lagers mit mehreren tausend Artikeln. Im folgenden werden wir uns daher auf den Einsatz im privaten Bereich beschränken.

Welche Befehle der C 16 zum Speichern und Laden von Dateien besitzt, will ich nun anhand eines kleinen Programms besprechen, das den Durchschnittsverbrauch eines Kraftfahrzeuges ermitteln soll. Um den Durchschnittsverbrauch zu berechnen, benötigt unser Programm Angaben über die gefahrenen Kilometer und den gesamten Benzinverbrauch. Der Gesamtverbrauch wird durch die Kilometerzahl geteilt und mit 100 multipliziert.

Beispiel:
Gefahrene Kilometer: 3435
Gesamtverbrauch: 273 Liter
Durchschnittsverbrauch pro 100 Kilometer:
273/3435*100=7,9 Liter

Selbstverständlich können wir den Durchschnittsverbrauch jederzeit auch ohne unseren C 16 ermitteln. Mit dem C 16 verringert sich der Aufwand hierzu jedoch erheblich: jedesmal wenn wir tanken, geben wir anschließend die Literzahl und den momentanen Kilometerstand ein. Alle nötigen Berechnungen soll unser Programm vornehmen.

```
200 rem durchschnittsverbrauch
210 ak=10000:rem anfangskilometerstand
220 :
230 input" anzahl getankter liter" ;li
240 sl=sl+li:rem summe liter
250 :
260 input" momentaner kilometerstand" ;km
270 sk=km-ak:rem summe kilometer
280 :
290 dv=sl/sk*100:rem verbrauch
300 print" durchschnittsverbrauch:" dv" 1/100 km"
310 :
320 goto 230:rem naechste eingabe
```

Programmablauf:

Wenn Sie dieses Programm benutzen, müssen Sie zuerst eine Programmzeile ändern: In Zeile 210 ist der Kilometerstand zu Beginn der Programmbenutzung festgelegt. Korrigieren Sie bitte den als Beispiel verwendeten Wert 10000 entsprechend Ihrem Tachostand.

Wenn Sie das Programm starten, werden Sie nach der jeweiligen Literzahl »li« gefragt. Alle von Ihnen nach Tankvorgängen eingegebenen Literzahlen werden addiert (Zeile 240). Die Variable »sl« stellt daher den bisherigen Gesamtverbrauch dar.

Anschließend werden Sie nach dem momentanen Kilometerstand gefragt. Die Summe der gefahrenen Kilometer ergibt sich aus der Differenz zwischen dem momentanen Kilometerstand »km« und dem Stand zu Beginn der Ermittlung des Verbrauchs »ak« (Zeile 270).

Wie besprochen wird nun der Durchschnittsverbrauch ermittelt, indem der Gesamtverbrauch »sl« durch die gefahrenen Kilometer »sk« geteilt und das Ergebnis mit 100 multipliziert wird (Zeile 290). Der in der Variablen »dv« enthaltene Durchschnittsverbrauch wird errechnet und auf dem Bildschirm ausgegeben.

Den Abschluß des Programms bildet ein Sprung zum Beginn der Abfragen (Zeile 320). Wenn Sie Ihren C 16 Tag und Nacht eingeschaltet lassen, wird das Programm geduldig warten, bis Sie wieder tanken und erneut Literzahl und Kilometerstand eingeben.

Dieses Programm funktioniert zwar einwandfrei (probieren Sie's aus), hat jedoch den Nachteil, daß Ihr C 16 für keinen anderen Zweck mehr verwendbar ist. Sobald Sie das Programm unterbrechen und ein anderes Programm eingeben oder den Computer ausschalten, geht der Inhalt der Variablen »sl«, das heißt die Summe getankter Liter, verloren, und alle folgenden Berechnungen, die auf diesem Wert basieren, sind falsch!

Dieser Wert muß daher unbedingt nach jeder Programmbenutzung in einer Datei abgespeichert werden, und vor der eigentlichen Dateneingabe und Verbrauchsermittlung muß diese Datei in den Computerspeicher geladen werden.

Datenkanal öffnen:

Um Daten auf Band zu schreiben oder zu lesen, muß immer ein sogenannter »Datenkanal« geöffnet werden. Diesen Kanal kann man sich als Verbindung zwischen C 16 und Kassettenrecorder vorstellen, über den die Daten ausgetauscht werden. Der zugehörige Befehl besitzt folgende Syntax:

```
OPEN (LF),(GN),(SA)," (FN)"
```

LF: logische Filenummer, eine beliebige Zahl zwischen 1 und 255. Wenn im weiteren Programmablauf Daten geschrieben oder gelesen werden sollen, muß immer diese Filenummer angegeben werden.

GN: Gerätenummer. Sie gibt an, mit welchem Gerät (Drucker, Datasette, Floppy) Daten ausgetauscht werden sollen. Die Datasette besitzt die Gerätenummer 1, die Floppy die Gerätenummer 8.

SA: Sekundäradresse, im Fall der Datasette eine Null, Eins oder Zwei. Benötigt werden üblicherweise nur die Zahlen Null und Eins, wobei Null bedeutet, daß Daten vom Band gelesen werden sollen, und Eins, daß Daten auf das Band geschrieben werden.

FN: Filename (=Dateiname), der Name der betreffenden Datei. Die Angabe eines Filenamens ist sehr vorteilhaft, da mehrere Dateien auf einer Kassette enthalten sein können. Wenn wir eine bestimmte Datei lesen wollen, wird unser C 16 alle vorhergehenden Dateien überlesen, bis er zu jener Bandstelle kommt, an der die Datei beginnt, deren Filenamen wir im OPEN-Befehl angaben.

Beispiele:

```
OPEN 1,1,0," TESTDATEI"
```

Öffnet einen Datenkanal zwischen C 16 und Datasette (Gerätenr. 1). Die Richtung des Datenaustauschs verläuft von der Datasette zum C 16 (Sekundäradresse 0 = Daten lesen), die gesuchte Datei hat den Namen »TESTDATEI«.

```
OPEN 1,1,1," VERBRAUCH"
```

Öffnet einen Datenkanal zwischen C 16 und Datasette. Da als Sekundäradresse eine Eins angegeben wurde, sollen Daten auf das Band geschrieben werden. Die Datei erhält den Namen »VERBRAUCH«.

Daten auf Band schreiben:

Um ein Programm zu laden oder zu speichern genügt die Eingabe eines einzigen Befehls (»LOAD«/»SAVE«), weitere Angaben sind nicht erforderlich.

Die Arbeit mit Dateien ist erheblich aufwendiger, da wir unserem Computer angeben müssen, welche Daten abgespeichert oder eingelesen werden sollen. Die Befehlssyntax zum Schreiben von Daten auf Kassette:

```
PRINT#(LF),(DATEN)
```

LF: Logische Filenummer, die wir beim Öffnen des Datenkanals angaben.

DATEN: Variable, die die zu schreibenden Daten enthält oder unmittelbare Angabe der Daten.

Beispiele:

```
PRINT#1,A$
```

Schreibt den Inhalt der Variablen A$ auf Band (wenn zuvor ein Kanal zum Schreiben unter der log. Filenummer Eins geöffnet wurde).

```
PRINT#1,X
```

Schreibt den Inhalt der numerischen Variablen X auf Band.

```
PRINT#1," DIES IST EIN TEST"
```

Schreibt die Zeichenkette »DIES IST EIN TEST« auf Band.

Die Ähnlichkeit mit dem PRINT-Befehl, der der Datenausgabe auf dem Bildschirm dient, ist unverkennbar. Der einzige Unterschied besteht in der Angabe der logischen Filenummer, damit unser Computer die Daten nicht auf dem Bildschirm ausgibt, sondern über den bereits geöffneten Datenkanal an die Datasette schickt.

Daten vom Band lesen:

Viele werden nun bereits ahnen, wie der entsprechende Befehl zum Einlesen von Daten lautet: INPUT#. Die genaue Syntax:

```
INPUT#(LF),(VARIABLE)
```

LF: Logische Filenummer, die wir beim Öffnen des Datenkanals angaben.

DATEN: Variable, in der die zu lesenden Daten übergeben werden sollen.

Beachten Sie auch beim INPUT#-Befehl die Analogie zum INPUT-Befehl, der eine über die Tastatur vorgenommene Eingabe einer Variablen zuweist. INPUT# erfüllt die gleiche Aufgabe für Eingaben von einem Peripheriegerät (Datasette, Floppy).

Beispiele:

```
INPUT#1,A$
```

Liest eine Zeichenkette und übergibt sie in der Variablen A$.

```
INPUT#1,X
```

Liest eine Zahl und übergibt sie in der Variablen X.

Datenkanal schließen:

Wenn eine Datei komplett in den Computerspeicher gelesen oder auf Band geschrieben wurde, muß der Datenkanal wieder geschlossen werden. Vor allem nach dem Schreiben einer Datei kommt es zu Problemen, wenn dieser Vorgang nicht ordnungsgemäß durchgeführt wird: das spätere Einlesen der Datei ist dann nicht mehr möglich, unsere Daten sind verloren! Die Syntax des sogenannten »CLOSE«-Befehls:

```
CLOSE (LF)
```

LF: logische Filenummer, unter der der Kanal geöffnet wurde.

Beispiele:

```
CLOSE 1
```

Schließt einen unter der log. Filenummer Eins geöffneten Datenkanal.

```
CLOSE 7
```

Schließt einen unter der log.Filenummer Sieben geöffneten Datenkanal.

Halten wir fest: Um Daten vom Band zu lesen oder auf Band zu schreiben, muß zuerst ein Datenkanal mit dem OPEN-Befehl geöffnet werden. Mit Hilfe der Sekundäradresse können wir angeben, ob Daten gelesen oder geschrieben werden sollen. Lesen und schreiben zugleich ist nicht möglich!

Anschließend können beliebige Daten mit dem PRINT#-Befehl in eine Datei geschrieben oder mit dem INPUT#-Befehl gelesen werden. Beim Schreiben und Lesen von Daten können sowohl numerische als auch Stringvariablen verwendet werden.

Den Abschluß bildet das ordnungsgemäße Schließen der Datei mit dem CLOSE-Befehl, das vor allem nach Schreibvorgängen keinesfalls vergessen werden darf!

Wir besitzen nun das nötige Wissen, um unser Programmbeispiel vervollständigen zu können.

Datei speichern:

```
400 open 1,1,1," verbrauch" :rem datenkanal oeffnen
410 print#1,sl:rem gesamte literzahl speichern
420 close 1:rem kanal schliessen
```

In Zeile 400 wird ein Datenkanal zur Datasette zum Schreiben unter der logischen Filenummer Eins geöffnet (log. Filenummer 1, Gerätenr.1, Sekundäradresse 1). Die Datei erhält den Namen »VERBRAUCH«.

In Zeile 410 wird der Inhalt der Variablen »sl«, das heißt die gesamte Literzahl, in die Datei geschrieben.

In Zeile 420 wird der unter der log. Filenummer Eins geöffnete Datenkanal unter Angabe der gleichen Filenummer wieder geschlossen.

Datei einlesen:

```
100 open 1,1,0," verbrauch" :rem datenkanal oeffnen
110 input#1,sl:rem gesamte literzahl einlesen
120 close 1:rem kanal schliessen
```

Zeile 100: Im OPEN-Befehl zum Einlesen der Datei muß unbedingt der gleiche Dateiname angegeben werden, unter dem die Datei gespeichert wurde. Wichtig ist außerdem die Angabe einer Null als Sekundäradresse (= Lesen von Daten).

In Zeile 110 wird die Literzahl in die Variable »sl« eingelesen und in Zeile 120 der Datenkanal wieder geschlossen.

Vollständiges Programmlisting:

```
90 rem datei einlesen
100 open 1,1,0," verbrauch" :rem datenkanal
oeffnen
110 input#1,sl:rem gesamte literzahl einlesen
120 close1:rem kanal schliessen
130 :
200 rem durchschnittsverbrauch
210 ak=10000:rem anfangskilometerstand, bitte
individuell aendern
220 :
230 input" anzahl getankter liter" ;li
240 sl=sl+li:rem summe liter
250 :
260 input" momentaner kilometerstand" ;km
270 sk=km-ak:rem summe kilometer
280 :
290 dv=sl/sk*100:rem verbrauch
300 print" durchschnittsverbrauch:" dv" 1/100 km"
310 print" bitte zurueckspulen und taste druecken"
320 getkey a$
380 :
390 rem datei speichern
400 open 1,1,1," verbrauch" :rem datenkanal
oeffnen
410 print#1,sl:rem gesamte literzahl speichern
420 close1:rem kanal schliessen
```

Tippen Sie nun das vollständige Programm ab, ändern Sie Zeile 210 (Anfangskilometerstand) entsprechend Ihrem Tachostand ab, legen Sie eine leere Kassette ein und speichern Sie das Programm. Lassen Sie die Kassette bitte ein-

gelegt und stellen Sie das Zählwerk der Datasette auf »000«. Sie dürfen das Programm bei der ersten Benutzung keinesfalls mit »RUN« starten, da der C 16 sonst versuchen würde, die noch nicht vorhandene Datei »VERBRAUCH« zu laden (Zeilen 100-120)!

Starten Sie das Programm mit »RUN 200«, und geben Sie zum Testen beliebige - aber einigermaßen sinnvolle - Werte für die Anzahl getankter Liter und den momentanen Kilometerstand ein. Der bisherige Durchschnittsverbrauch wird nun ermittelt und auf dem Bildschirm ausgegeben.

Anschließend wird die Datei gespeichert. Sie erhalten die Aufforderung die Kassette zum Dateianfang zurückzuspulen und anschließend eine beliebige Taste zu drücken. Nach diesem Vorgang erscheint auf dem Bildschirm die bekannte Meldung »PRESS PLAY AND RECORD ON TAPE«. Drücken Sie beide Tasten gleichzeitig. Die Datei, die in unserem Programmbeispiel nur aus der Variablen »sl« besteht, wird gespeichert.

Spulen Sie nun zurück bis zum Zählerstand »000«, das heißt bis zu jener Bandposition, ab der die Datei gespeichert wurde, und starten Sie das Programm mit »RUN«. Sie erhalten die Aufforderung »PRESS PLAY ON TAPE«. Drücken Sie die entsprechende Taste der Datasette. Nachdem die Datei geladen wurde, ist in der Variablen »sl« somit die gesamte Literzahl enthalten. Nun können Sie neue Werte für die Anzahl der getankten Liter und den Kilometerstand eingeben. Das Programm ermittelt wiederum den Durchschnittsverbrauch und gibt ihn aus.

Spulen Sie nun wiederum bis zum Dateianfang zurück, bevor Sie die Tasten »PLAY« und »RECORD« betätigen. Die Datei wird erneut abgespeichert, wobei die alte Datei überschrieben wird.

Diese Vorgehensweise ist typisch für den Umgang mit Programmen, die sequentielle Dateien verwalten. Zusammenfassung der einzelnen Schritte:

1. Speichern Sie die Datei unmittelbar hinter dem Programm auf die Kassette. Wenn die Datei nach dem Programmstart eingelesen werden soll, ist kein Spulen nötig, da sich das Band automatisch am Dateianfang befindet.

2. Stellen Sie das Kassettenzählwerk nach dem Einlesen des Programms auf »000« zurück, um den Dateianfang zu kennzeichnen.

3. Wenn die Datei wieder gespeichert werden soll, spulen Sie zur Position »000« zurück, das heißt zum Dateianfang. Die alte Datei wird beim Speichern durch die neuen Daten überschrieben werden.

Damit Sie mit den beschriebenen Befehlen vertrauter werden, stelle ich ein weiteres Programmbeispiel zum Lesen und Schreiben von Daten vor:

```
100 a$=" open = oeffnen eines datenkanals"
110 b$=" print#= schreiben von daten"
120 c$=" input#= lesen von daten"
130 d$=" close = datenkanal schliessen"
140 :
150 rem datei schreiben
160 open 1,1,1," test"
170 print#1,a$
180 print#1,b$
190 print#1,c$
200 print#1,d$
210 close1
220 :
230 print" bitte zum dateianfang zurueckspulen"
240 print" und danach eine beliebige taste
druecken"
250 getkey a$
260 :
270 rem datei einlesen
280 open 1,1,0," test"
290 input#1,a$
300 input#1,b$
310 input#1,c$
320 input#1,d$
330 close1
340 :
350 printa$
360 printb$
370 printc$
380 printd$
```

Dieses Programm erzeugt mehrere Stringvariablen, schreibt sie in eine Datei, fordert Sie anschließend auf, zum Dateianfang zurückzuspulen und liest die Datei wieder ein.

Häufige Fehler beim Umgang mit sequentiellen Dateien:

1. Es kann vorkommen, daß Ihnen der C 16 einen »SYNTAX ERROR IN ...« ausgibt, Sie jedoch in der betreffenden Zeile keinen Fehler entdecken können. Meist handelt es sich um einen falsch eingegebenen PRINT#-Befehl. Wie Ihnen vielleicht bekannt ist, können Sie PRINT-Befehle mit dem Fragezeichen abkürzen, zum Beispiel: 10?»HALLO«. Beim LISTEN der Zeile erscheint der vollständige Befehl: 10PRINT»HALLO«. Der PRINT#-Befehl darf nicht (!) auf diese Weise abgekürzt werden. Dieser Fehler ist schwer zu entdecken, da beim LISTEN der betreffenden Zeile auch der auf diese Weise eingegebene PRINT#-Befehl vollständig und scheinbar korrekt erscheint, zum Beispiel: 10PRINT#1,A$. Die einzige Möglichkeit, diesen Fehler zu beheben, ist die korrekte Neueingabe des Befehls, obwohl Sie im Listing des Programms keinen Unterschied entdecken werden. Ich gebe zu, daß dies schwer verständlich ist, und rate Ihnen daher, es selbst auszuprobieren.

2. »FILE DATA ERROR«: Dieser Fehler tritt meist auf, wenn beim Lesen einer Datei nicht die gleiche Reihenfolge wie beim Schreiben der Daten eingehalten wird. Wurden zum Beispiel folgende Variablen auf Kassette gespeichert:

```
10 print#1,a$
20 print#1,b
30 print#1,c$
```

und sollen sie wie folgt eingelesen werden:

```
10 input#1,b
20 input#1,a$
30 input#1,c$
```

dann erhalten Sie die erwähnte Fehlermeldung, da mit dem ersten INPUT#-Befehl eine numerische Variable eingelesen werden soll (»b«), sich an dieser Stelle der Datei jedoch alphanumerische Daten befinden, nämlich der Inhalt der Variablen »a$«. Denken Sie daran, daß die Daten einer sequentiellen Datei der Reihe nach gelesen werden müssen und ein Überspringen von Daten nicht möglich ist!

3. Das Handbuch enthält eine Bemerkung, die viel Ärger bereiten kann. Angeblich soll es möglich sein, in einem PRINT#-Befehl eine Liste mehrerer Daten anzugeben und diese durch Kommata oder Semikolons zu trennen, zum Beispiel:

```
10 print#1,a$,b$,c$
```

Es soll nun möglich sein, diese Daten mit einem INPUT#-Befehl, in dem die gleichen Trennzeichen verwendet werden, wieder einzulesen:

```
10 input#1,a$,b$,c$
```

Diese Möglichkeit wäre zwar sehr komfortabel, hat jedoch ihre Tücken: Beim Einlesen kann der INPUT#-Befehl die einzelnen Daten nicht unterscheiden. Alle drei Daten (a$,b$,c$)

werden in die erste im INPUT#-Befehl aufgeführte Variable »a$« eingelesen, die danach den Inhalt »a$+b$+c$« besitzt. Der C 16 ist nicht mehr in der Lage, das Ende einer der drei Zeichenketten zu erkennen, und behandelt alle drei wie eine einzige Zeichenkette. Verwenden Sie daher bitte zum Schreiben von Daten auf Kassette je einen PRINT#-Befehl zum Speichern einer Variablen.

Der GET#-Befehl:

Besondere Erwähnung verdient der Befehl GET#(LF),(VARIABLE). Mit ihm lassen sich mehrere Unzulänglichkeiten des INPUT#-Befehls ausmerzen. Mit dem PRINT#-Befehl können Zeichenketten mit einer Länge von maximal 255 Zeichen gespeichert werden. Der INPUT#-Befehl kann jedoch nur Zeichenketten einlesen, deren Länge höchstens 80 Zeichen beträgt (probieren Sie's aus!).

Mit dem Befehl GET# können Zeichenketten beliebiger Länge eingelesen werden. Dieser Befehl liest ein einziges Zeichen aus einer Datei ein und übergibt es in der angegebenen Variablen. Eine Zeichenkette kann auf folgende Weise komplett eingelesen werden:

```
10 open 1,1,0," test"
20 get#1,a$
30 if a$=chr$(13) then print zk$:end
40 zk$=zk$+a$
50 goto 20
```

Um dieses Programm zu verstehen, muß man wissen, daß nach dem PRINT#-Befehl ein Sonderzeichen auf das Band geschrieben wird. Dieses Sonderzeichen hat den ASCII-Code 13 (zum ASCII-Code siehe C 16-Handbuch) und dient als Trennzeichen zwischen den einzelnen Daten, die in einer Datei vorhanden sind. An diesem Trennzeichen erkennt unser C 16, ob eine Zahl oder Zeichenkette bereits komplett eingelesen wurde.

Das Programmbeispiel geht ebenso vor: Es liest Zeichen für Zeichen aus der Datei »TEST« und verknüpft die einzelnen Zeichen zum String »zk$«. Wenn der ASCII-Code des gerade gelesenen Zeichens den Wert 13 hat (»asc(a$)«), handelt es sich um das Trennzeichen, und die Zeichenkette wurde komplett eingelesen (Zeile 30).

Verwenden Sie bitte folgendes Programm, um eine Zeichenkette auf Band zu schreiben:

```
10 open 1,1,1," test"
20 a$=" dies ist ein test"
30 print#1,a$
40 close1
```

Wenn Sie diese Minidatei gespeichert haben, können Sie die Zeichenkette »dies ist ein test« mit dem obigen Programm wieder einlesen. Der Nachteil des GET#-Befehls besteht darin, daß das Einlesen von Daten erheblich langwieriger ist als mit dem schneller arbeitenden INPUT#-Befehl. Sollten Sie jedoch jemals das Problem sehr langer einzulesender Zeichenketten haben, denken Sie an diesen GET#.

Sequentielle Dateien mit der Floppy:

Wenn Sie zu den glücklichen Besitzern einer Floppy gehören, so können Sie selbstverständlich ebenfalls sequentielle Dateien verwalten. Außer den sequentiellen stehen Ihnen zudem noch die erwähnten Direktzugriffsdateien zur Verfügung. Um den doch etwas aufwendigeren Umgang mit Direktzugriffsdateien zu lernen, empfehle ich Ihnen die Lektüre einschlägiger Zeitschriften und Bücher, da eine ausführliche Beschreibung dieser Dateiformen den Rahmen dieses Artikels sprengen würde.

Zur Verwaltung sequentieller Dateien mit der Floppy können Sie von dem bereits Beschriebenen ausgehen, wenn Sie folgende Unterschiede zur Datasette beachten:

1. Im OPEN-Befehl müssen Sie selbstverständlich die Gerätenummer der Floppy (acht) angeben.
2. Verwenden Sie als Sekundäradresse bitte nicht die Zahl 15, da diese Sekundäradresse einem besonderen Zweck vorbehalten ist.
3. Die Angabe, ob Daten gelesen oder in eine Datei geschrieben werden sollen, erfolgt nicht mehr in der Sekundäradresse, sondern im Filenamen, dem die Zeichenkette »,s,w« für Schreiben oder »,s,r« für Lesen folgen muß.
4. Um eine Datei zu überschreiben, müssen sich vor dem Dateinamen die beiden Zeichen »@:« befinden. Das Überschreiben geschieht nicht automatisch wie bei der Datasette.

Beispiele:

```
OPEN 2,8,2," TEST,S,R"
```

Öffnet die Datei »TEST« zum Lesen von Daten.

```
OPEN 2,8,2," TEST,S,W"
```

Öffnet die neue Datei »TEST« zum Schreiben von Daten.

```
OPEN 2,8,2,"@:TEST,S,W"
```

Öffnet die Datei »TEST« zum Schreiben von Daten. Eine unter diesem Namen bereits existierende Datei wird überschrieben.

Die Syntax aller übrigen beschriebenen Befehle bleibt unverändert. Zum Abschluß wünsche ich Ihnen viel Erfolg bei der Erstellung Ihrer eigenen Adress-, Haushaltskosten- und sonstigen Dateien, ob sich diese nun auf Kassette oder Diskette befinden mögen.

(Said Baloui/kn)



Fragen und Antworten zum C16 und VC 20

Hier sollen Fragen zum C16 und VC20, die sehr häufig gestellt wurden, umfassend beantwortet werden.

Maschinenprogramme kopieren?

Wie kopiert man beim VC 20 ein Maschinenprogramm im Steckmodulbereich ($A000) von einer Disk auf eine andere?

1. Speichererweiterung auf $A000 umadressieren.
2. Maschinenprogramm absolut laden (»LOAD"Name",8,1«), aber nicht starten.
3. Diskette wechseln und folgende Zeile im Direktmodus hintereinander eingeben:

```
POKE 43,0 : POKE 44,160 : POKE 45,0 :
POKE 46,192 : SAVE"Name",8
```

Vor dem weiteren Arbeiten mit dem Computer »NEW« eingeben.

VC 20-Programme auf C16?

Kann man Programme für den VC 20 auch auf dem C16 verwenden?

VC 20/C64-Programme laufen immer dann garantiert auf dem C16, wenn Sie keine PEEK-, POKE-, SYS- und WAIT-Befehle enthalten. In vielen Fällen handelt es sich bei POKE-Befehlen beim VC 20/C64 allerdings nur um einfache Dinge wie Bildschirmfarbe einstellen oder einen Ton erzeugen. Falls ein Listing nur wenige POKEs enthält, lohnt sich oft ein Versuch auf dem C16, bei dem man die POKE-, PEEK- und SYS-Befehle einfach fortläßt. Farbe und Sound kann man ja schließlich selbst noch ganz nach Belieben ins Programm einbauen.

Modulbereich nutzen?

Wie kann man beim VC 20 mit 24 KByte-Erweiterung den geänderten Zeichensatz in den Modulbereich ($A000 bis $BFFF) verlegen? Welcher Wert muß dazu in die Speicherstelle 36869 gePOKEt werden? Muß dazu auch der Bildschirmspeicher verlegt werden?

Die Bits 0 bis 3 aus Adresse 36869 (Register 5 des VIC) bestimmen die Lage des Zeichengenerators im Speicher und zwar nach einem recht komplizierten Schema, bei dem einige Bits der generierten effektiven Adresse immer auf Null sind. Leider gehören auch die Bits 13 und 14 der Zeichenspeicher-Adresse zu diesen Null-Bits. Dadurch ist der Speicherbereich ab $A000 (=1010 0000 0000 0000 binär) nicht als Zeichengenerator-Adresse einstellbar. Auch der Bildschirmspeicher läßt sich beim besten Willen nicht in diesen Bereich verlegen.

Speicherverlust durch Grafik?

Bei Verwendung hochauflösender Grafik verringert sich die Speicherkapazität des C16 um rund 10 KByte. Woran liegt das, und wie kann man das verhindern?

Damit die Grafik am Bildschirm sichtbar sein kann, muß sie irgendwo im Computer gespeichert sein. Jedem einzelnen Punkt am Bildschirm ist dabei ein Bit im Speicher zugeordnet. Nun hat der C16 eine Grafikauflösung von 360 x 200 Punkten, das sind insgesamt 72000 Punkte, die ebensovielen Speicherbits entsprechen. 8 Bit ergeben ein Byte, die 72000 Bildpunkte belegen also genau 8000 Byte im Speicher. Außerdem wird noch zusätzlicher Speicher benötigt, um die Farbinformationen für jede Bildschirmstelle festzuhalten, so daß insgesamt ungefähr 10 KByte benötigt werden, um überhaupt die Grafik einschalten zu können. Das läßt sich unter keinen Umständen verhindern. Abhilfe schafft hier nur eine Speichererweiterung, bei der ein Verlust von 10 KByte nicht so schwer ins Gewicht fällt wie bei den mageren serienmäßigen 16 KByte Speicher.

Probleme mit Speichererweiterung?

Spiele, die für den VC 20 mit 3 KByte-Erweiterung geschrieben wurden, laufen leider nicht mit einer 8- oder 16 KByte-Erweiterung. Gibt es einen Trick, um sich dennoch das ständige Umstecken der Erweiterungen zu ersparen?

Abhilfe schafft hier zum Beispiel die Verwendung einer Modulbox, in die mehrere Steckmodule gleichzeitig eingesteckt werden können. Aber es geht auch ohne Modulbox: Mit der folgenden kurzen Routine lassen sich die meisten Grundversions- oder +3 KByte-Programme auch mit einer 8 (16) KByte-Erweiterung laden und ausführen:

```
POKE 648,30 : SYS 64821
POKE 4096, 0 : POKE 44, 16 : NEW
```

Da der Bildschirmspeicher durch die kleine Routine an der gleichen Stelle wie in der Grundversion liegt, sind die meisten Grundversions-Programme direkt lauffähig.

63299 Bytes free beim VC 20?

Die Befehlsfolge »POKE 56,255 : SYS 58234« soll beim VC 20 einen freien Speicherplatz von über 63000 Bytes ergeben. Sind diese 63299 Bytes wirklich frei zum Programmieren und wird dadurch auch keine andere Funktion beeinflußt?

Na, da hat Sie aber wirklich einer an der Nase herumgeführt. Der POKE-Befehl setzt zwar (allerdings nur rein rechnerisch) die Speichergrenze für den Basic-Interpreter nach oben, aber sonst tut sich überhaupt nichts, denn selbstverständlich kann dort, wo einfach kein Speicher ist, nicht durch einen simplen POKE-Befehl welcher aus dem Nichts erzeugt werden. Im Gegenteil, wenn Sie längere Programme laden oder mit Variablen arbeiten, dann kann es sogar zum Absturz des Computers kommen, weil dieser verzweifelt versucht, Programm oder Variablen in nicht vorhandenen Speicherbereichen abzulegen. Das kann natürlich nicht gutgehen.

Der SYS-Befehl hinter dem POKE ist übrigens reine Augenwischerei, denn der bewirkt nur einen Sprung in die NMI-Routine, also dasselbe, was beim Drücken der Tastenkombination RUN/STOP-RESTORE passiert.

Assembler und andere Mysterien

Was ist ein Assembler, was ein Disassembler und was versteht man unter einem Maschinensprache-Monitor?

Wie Sie vielleicht wissen, versteht der im VC 20 arbeitende 6502-Prozessor (wie auch der 7501-Prozessor im C 16) die Programmiersprache Basic nicht direkt, sondern nur eine sehr viel einfacher (für den Prozessor) aufgebaute Sprache, eben die sogenannte Maschinensprache. Sie können mit Ihrem Computer nur deshalb in Basic arbeiten, weil der VC 20/C 16 ein Programm fest gespeichert hat, das einen »Basic-Prozessor« simuliert. Der Prozessor selbst versteht also kein Basic, aber dieses Programm, der sogenannte »Basic-Interpreter«, simuliert einen viel leistungsfähigeren Prozessor, der Basic verstehen kann. Diese Simulation kostet natürlich Zeit, viel Zeit sogar. Daher benötigt der Computer für die Abarbeitung eines Basic-Befehls im Schnitt einige tausendstel oder sogar hundertstel Sekunden, ein Befehl in Maschinensprache dagegen nur wenige millionstel Sekunden.

Natürlich können Sie Ihren VC 20 oder C 16 auch direkt in Maschinensprache programmieren – das ist ja quasi seine Muttersprache. Allerdings geht das von Basic aus nicht ganz so einfach. Sie müssen sich klarmachen, daß ja bereits die Abfrage der Tastatur oder das Blinken des Cursors durch recht komplexe Programme in Maschinensprache realisiert werden, auch wenn Sie nichts davon bemerken. Es gibt aber für die Programmierung in Maschinensprache spezielle Hilfsprogramme. In der einfachsten Form handelt es sich dabei um Maschinensprache-Monitore, mit denen man – vereinfacht gesagt – kleine Maschinenprogramme gleich in der dem Prozessor geläufigen Zahlenform eingeben kann.

Dieses Programmieren in Zahlen ist zwar für die Maschine sehr angenehm, aber weniger für den Menschen. Daher wurden zu jedem Maschinenbefehl sogenannte »Mnemonics«, also leicht zu merkende Abkürzungen für die Wirkungsweise der Befehle, geschaffen. Um diese Mnemonics in das der Maschine geläufige Zahlenformat zu übersetzen, gibt es wiederum andere Programme – die Assembler.

Ein Disassembler schließlich tut das Gegenteil vom Assembler: Mit ihm können fertige Maschinenprogramme wieder in die Assembler-Mnemonics zurückübersetzt werden.

Kein Speicherplatz mehr?

Mein Datenverwaltungs-Programm für den C 16 ist durch eine Reihe von nachträglichen Verbesserungen so umfangreich geworden, daß kein Platz mehr für die Daten im Speicher ist. Was kann ich tun?

Als erstes: Ruhe bewahren. Als zweites bieten sich zwei Methoden an, eine einfache und eine schwierige. Zunächst die Schwierige: Es ist eine altbekannte Programmierer-Erfahrung, daß Programme insbesondere durch nachträgliche Änderungen sehr umfangreich werden. Sie könnten also versuchen, ein neues Konzept für Ihr Programm zu entwerfen, das alle Änderungen berücksichtigt, aber von Grund auf neu programmiert wird. Häufig läßt sich dadurch einiges an Speicherplatz gegenüber »wilden« Änderungen einsparen. Aber auch die einfache Methode soll angesprochen werden: Wenn der Speicher nicht mehr reicht, reicht häufig mehr Speicher. Besorgen Sie sich also doch ganz einfach eine Speichererweiterung für Ihren C 16. Mit der in diesem Sonderheft vorgestellten 64 KByte-Erweiterung von Kingsoft meldet sich der C 16 nach dem Einschalten zum Beispiel mit »60671 Bytes free«. Damit dürften dann sogar noch weitere kleine Erweiterungen an Ihrem Programm zu realisieren sein...

VC 20 zum C 64 aufrüsten?

Kann man den VC 20 zum C 64 aufrüsten?

Der VC 20 kann nicht in dem Sinne zum C 64 aufgerüstet werden, daß anschließend darauf die C 64-Software laufen würde (es sei denn, man baut eine komplette C 64-Platine in das VC 20-Gehäuse ein). Speichermäßig ist ein Ausbau natürlich möglich, die C 64-Programme laufen dann allerdings trotzdem nicht richtig, denn die gesamte Hardware des VC 20 weist große Unterschiede zum C 64 auf.

Wer C 64-Programme auf dem VC 20 laufen lassen will, sollte sich den Artikel »C 64-Programme für C 16 und VC 20« ansehen, den wir in diesem Sonderheft veröffentlicht haben.

Programmunterbrechung bei Druckerausgabe?

Bei der Druckerausgabe erscheint oft ein »DEVICE NOT PRESENT ERROR«, obwohl der Drucker angeschlossen und eingeschaltet ist. Was kann man dagegen tun?

Dieser Fehler tritt vergleichsweise oft bei Druckern auf, die direkt am seriellen Bus angeschlossen sind. Die Ursache hierfür liegt in der Konzeption des seriellen Busses begründet. Abhilfe: Statt mit »PRINT #« mit CMD arbeiten, dann tritt der Fehler nur noch äußerst selten oder überhaupt nicht mehr auf.

Sprites beim C 16?

Ist es möglich, beim C 16 Sprites im Textmodus softwaremäßig zu simulieren?

Der C 16 besitzt die Shape- beziehungsweise Sprite-Fähigkeit nur im Grafik-Modus, weil beim TED-Chip (der Chip, der beim C 16 für die Video-Ausgabe zuständig ist) Sprites nicht hardwaremäßig vorgesehen sind. Falls man aus Speicherplatzgründen auf den Grafik-Modus verzichten will oder muß, dann kann man sich aber recht gut wie beim VC 20 mit einem selbstdefinierten Zeichensatz behelfen, der nur etwa zwei KByte vom knappen Speicher beansprucht und dennoch schon beachtliche Grafikmöglichkeiten, gerade für den Spielebereich, bietet. Mehr Informationen zu diesem Thema bietet Ihnen der Beitrag »Schnelle Spielegrafik für den C 16« in diesem Sonderheft.

Wie kompatibel sind 6502 und 7501?

Der VC 20 hat als Prozessor den altbekannten 6502 eingebaut, der C 16 dagegen den 7501. Gibt es Unterschiede im Befehlssatz beider Prozessoren?

Der 7501 ist vom Befehlssatz her identisch mit dem 6502, so daß es zumindest von daher keine Probleme etwa beim Umschreiben von Maschinenprogrammen des VC 20 für den C 16 gibt. Allerdings ist es denkbar, daß sich einige der sogenannten »undefinierten Opcodes« beider Prozessoren voneinander unterscheiden. Bei diesen undefinierten Opcodes handelt es sich um vom Hersteller nicht vorgesehene und nicht dokumentierte Maschinenbefehle, von deren Benutzung abgeraten werden muß.

DFÜ mit C16?

Ist es möglich, mit dem C16 oder Plus 4 Datenfernübertragung zu betreiben, und was benötigt man dazu?

Zur Datenfernübertragung, egal mit welchem Computer, brauchen Sie grundsätzlich eine serielle Schnittstelle (RS232C) mit entsprechender Treibersoftware und einen Akustikkoppler. Die Wahl des Akustikkopplers ist unabhängig vom verwendeten Computermodell, da der Koppler immer über eine zwischengeschaltete serielle Schnittstelle betrieben wird. Über serielle Schnittstellen zum C16 informiert Sie Ihr Commodore-Händler, der Plus 4 hat bereits eine solche eingebaut. Der eigentlich kritische Punkt ist die notwendige Betriebssoftware für den Akustikkoppler. Sie brauchen ein Terminalprogramm, um Daten über Akustikkoppler mit anderen Computern austauschen zu können.

Leider tut sich in dieser Hinsicht bei den Herstellern von Terminalprogrammen nichts in Richtung C16 oder Plus 4. Der einzig gangbare Ausweg scheint das Selberschreiben zu sein – gegebenenfalls nach Lektüre eines oder mehrerer Bücher über DFÜ.

C16 statt C64?

Ich besitze seit geraumer Zeit einen C64, bin aber einerseits mit dem sehr mageren Basic unzufrieden, andererseits reicht auch der Basic-Speicher des C64 für meine selbstprogrammierten Anwendungen so langsam nicht mehr aus. Ich überlege nun schon seit einiger Zeit, ob sich der Umstieg auf den C16 lohnt, eventuell mit 64 KByte Speichererweiterung.

Der C16 ist durch sein wirklich umfangreiches und komfortables Basic 3.5 und aufgrund seines sehr niedrigen Preises eigentlich ein idealer Computer für Leute, die selbst in Basic programmieren wollen, was man vom C64 nicht sagen kann. Dessen Stärke ist ganz klar das Spielen, aber wegen seines Minimal-Basics eignet er sich kaum für den etwas ernsthafteren Hobby-Programmierer. Das soll natürlich nicht heißen, daß man mit dem C64 keine vernünftigen Programme selbst entwickeln kann, es heißt nur, daß es schwieriger ist, länger dauert und ohne Maschinensprachekenntnisse in vielen Fällen (Grafik) sogar fast unmöglich ist, leistungsfähige Basic-Programme auf dem C64 zu entwerfen.

Der große Schwachpunkt beim C16 ist eindeutig der mit 16 KByte völlig unterdimensionierte Speicher. Doch das läßt sich mittels Speichererweiterung leicht beheben. Ein einfaches Rechenexempel: Der Computer kostet um die 150 Mark, eine Speichererweiterung auf 64 KByte kostet knapp 200 Mark, und schon hat man einen leistungsfähigen Computer mit über »60000 Bytes free« zum Programmieren. Das ist um einiges billiger, als ein C64 mit schlechtem Basic und 38 KByte frei für Basic-Programme. Allerdings soll nicht verschwiegen werden, daß die C16-Lösung wirklich nur dem eingefleischten Selbst-Programmierer empfohlen werden kann. Mit der Wahl eines C16 verzichtet man nämlich auf das größte Softwareangebot, das es auf der Welt überhaupt für einen einzelnen Computer gibt, nämlich auf das breite Spektrum der C64-Software, das von brandheißen Spielen bis hin zu Textverarbeitung und Programmiersprachen reicht.

Ob sich der Umstieg vom C64 zum C16 oder umgekehrt lohnt, muß jeder für sich selbst entscheiden. Wer einmal mit dem komfortablen 3.5 Basic programmiert hat, wird sich nur schwer an die Programmierung des C64 gewöhnen, wer andererseits einmal beispielsweise »Summer Games« oder »Elite« auf dem C64 gespielt hat, wird vermutlich in dieser Hinsicht von allen anderen 8-Bit-Computern enttäuscht sein.

Probleme beim »Fensterln«

In meinem Dateiprogramm für den C16 ist es nicht möglich, die Position des gerade definierten Windows in einer Variablen WINDOW zu speichern. Ich erhalte immer nur einen »SYNTAX ERROR«. Bei Verwendung eines anderen Variablennamens, zum Beispiel FENSTER, gibt es keinen Fehler. Woran liegt das?

Sie verwenden in Ihrem Programm lange Variablennamen, was zwar einerseits die Übersichtlichkeit und Lesbarkeit Ihres Programms erhöht, andererseits aber auch gefährlich sein kann, wie Ihr Beispiel zeigt. Der Variablenname WINDOW enthält nämlich das Basic-Schlüsselwort DO. Während der Abarbeitung einer Programmzeile, die den Variablennamen WINDOW enthält, gerät der C16 daher arg ins Schleudern. Während er versucht, den kompletten Namen der Variablen zu erkennen und ihn sich zu merken, stößt er unvermittelt nach den Buchstaben W, I und N auf den Befehl »DO«, mit dem er an dieser Stelle gar nichts anfangen kann und sich nicht anders als mit einem »SYNTAX ERROR« zu helfen weiß.

Wenn Sie also in Ihren Programmen lange Variablennamen verwenden wollen, dann sollten Sie sehr genau prüfen, ob in diesen Namen etwa ein reserviertes Basic-Wort enthalten ist. Sie können das sehr einfach testen, indem Sie einfach PRINT und den Variablennamen eingeben. Antwortet der C16 mit »0«, dann ist alles ok.

Aber auch die »normalen« Variablennamen aus zwei Buchstaben sind nicht ohne Tücke. Folgende »kurze« Variablennamen sind ebenfalls nicht erlaubt: ST, TI, TI$, DS, DS$, ER, EL, GO, TO, DO, IF, OR.

Fehler durch Dimensionierung?

Wenn ich bei meinem C16 im Direktmodus eingebe »DIM A(2000,4)«, dann erhalte ich sofort die Fehlermeldung »OUT OF MEMORY«. Was mache ich falsch?

Die Fehlermeldung besagt ganz einfach, daß Ihr C16 nicht genug Speicher hat, um den Befehl auszuführen. Wenn es Ihnen ein Trost ist, auch der C64 gibt bei diesem Befehl die gleiche Fehlermeldung aus. Mit »DIM A(2000,4)« veranlassen Sie den Computer, ein Zahlenfeld namens »A« mit 2001 Zeilen (0 bis 2000) zu je 5 Spalten (0 bis 4) anzulegen. Das gesamte Feld enthält dann also 2001 x 5 = 10005 Elemente. Das allein sieht noch gar nicht so schlimm aus, aber überlegen wir weiter. Der C16 benötigt 5 Byte Speicherplatz für jeden Zahlenwert, den er sich merken soll. Das Feld A enthält 10005 Elemente, benötigt also 50025 Bytes Speicher! Da der C16 nach dem Einschalten nur 12277 Byte frei hat (der C64 genau 38911), ist einfach nicht genug Speicher vorhanden, um den DIM-Befehl ausführen zu können.

Sie können bei Feldern einiges an Speicherplatz sparen, wenn Sie Integer-Zahlen verwenden, denn Integer-Werte benötigen zur Speicherung nur zwei Byte pro Zahl. Allerdings können Integer-Variablen nur ganzzahlige Werte im Bereich zwischen -32768 und 32768 annehmen. Die Anweisung »DIM A%(2000,4)« benötigt beispielsweise »nur« 20010 Byte Speicher statt 50025 bei Verwendung normaler Fließkommazahlen (vereinfacht: Zahlen mit Werten hinter dem Komma). Da aber auch das für den C16 noch zuviel ist, hilft nur noch eines: Sie müssen versuchen, mit weniger Feldelementen auszukommen. Die Anweisung »DIM A% (1000,5)« packt der C16 zum Beispiel lässig (anschließend sind noch über 2000 Byte frei für ein Programm, das mit diesem Feld arbeitet).

Maschinensprache mit dem C16

Möchten Sie Maschinenprogramme für den C16 schreiben? Wenn ja, ist dieser Artikel genau das richtige. Wo Sie welche Betriebssystem-Einsprungadressen finden und wie Sie sie anwenden, wird hier ausführlich beschrieben.

In diesem Artikel soll besprochen werden, welche Besonderheiten bei der Programmierung des C16 in Maschinensprache zu beachten sind, welche Zeropage-Adressen unbenutzt sind und zum Beispiel für Pointer (indirekte Adressierung) verwendet werden können, wie die Betriebssystemroutinen benutzt werden und was bei der Einbindung von Maschinenroutinen in Basic-Programme beachtet werden muß.

Alle Beispielprogramme wurden möglichst einfach gehalten und unter Verzicht auf jegliche »Tricks« erstellt. Dieser Artikel kann jedoch keinesfalls ein Maschinensprache-Lehrgang sein oder einen solchen ersetzen. Ich muß im folgenden davon ausgehen, daß der Leser Grundkenntnisse über den Prozessor des C16, dessen Befehle und die verschiedenen Adressierungsarten besitzt.

Nehmen wir an, daß Sie Assemblergrundkenntnisse besitzen oder sich vor dem Weiterlesen zuerst auf den Kurs »Assembler ist keine Alchimie« (64'er Sonderheft 8/85 »Assembler«) stürzen, und fangen an.

Benutzung des integrierten Monitors:

Die in diesem Artikel beschriebenen Programmbeispiele sollten der Einfachheit halber mit dem eingebauten Maschinensprache-Monitor des C16 eingegeben werden. Da im Bedienungshandbuch leider jede Beschreibung des Monitors fehlt, erläutere ich kurz die wichtigsten Befehle (alle Zahlen müssen im Hexadezimalsystem eingegeben werden!) :

MONITOR = Start des Monitors

X = Verlassen des Monitors

S"(FILE)",(GERÄT),(START),(ENDE+1) = Speichern eines Speicherbereichs (Programms), zum Beispiel: `S"TEST",1,1000,2000` (Speichern von $1000 bis $1FFF unter dem Namen »TEST« auf Datasette) oder: `S"TEST",8,2000,2400` (Speichern von $2000 bis $23FF auf Diskette).

L"(FILE)",(GERÄT) = Laden eines Programms, zum Beispiel: `L"TEST",1` oder: `L"TEST",8`

.(ADRESSE) (MNEMONIC) = Assemblieren, zum Beispiel: `.1000 LDA #$1F`

> (ADRESSE) (BYTES) = Bytes eingeben, zum Beispiel: `>1000 1F 10 2B`

D (ANFANG) (ENDE) = Speicherbereich disassemblieren, zum Beispiel `D 1000 2000`

M (ANFANG) (ENDE) = Speicherbereich anschauen, zum Beispiel `M 1000 2000`

T (ANFANG) (ENDE) (ADRESSE) = Speicherbereich zu neuer Adresse verschieben, zum Beispiel: `T 1000 2000 3000`

Freie Zeropage-Adressen

Das Programmieren in Assembler ist nicht möglich ohne Kenntnis wichtiger Adressen und Speicherbereiche eines Computers. Der Prozessor des C16 bietet die sehr elegante indirekte Adressierung. Wie Sie wissen, ist es jedoch zur Verwendung dieser Adressierungsart notwendig, Pointer in der Zeropage des Computers abzulegen. Da – unabhängig von der indirekten Adressierung – die Zeropage die schnellste Form der Adressierung gestattet, die sogenannte »Zeropage-Adressierung«, wird man sie in zeitkritischen Routinen (Grafik!) ausgiebig verwenden, um möglichst viele Variablen darin abzulegen, auf die häufig zugegriffen werden muß. Der erste Schritt beim Kennenlernen des C16 besteht somit darin, herauszubekommen, welche Speicherplätze in der Zeropage unbenutzt sind und für eigene Zwecke zur Verfügung stehen.

Zuerst wäre der Bereich $D8 bis $E8 zu nennen. Dieser Bereich ist ungenutzt und steht sogar speziell für Anwendungssoftware zur Verfügung. In diesem Bereich können Sie alle Variablen ablegen, die ständig unverändert zur Verfügung stehen sollen. Alle weiteren Bereiche, die ich vorstelle, werden vom Basic-Interpreter beziehungsweise vom Betriebssystem benutzt und sind daher nicht unbedingt vorm Überschreiben geschützt.

Ein Beispiel hierfür ist der Bereich $61 bis $70, die Fließkommaakkumulatoren eins und zwei. Dieser recht große zusammenhängende Bereich wird vom Basic-Interpreter für Berechnungen aller Art benutzt, übrigens keineswegs nur für Berechnungen mit reellen Zahlen, wie man aufgrund des Namens annehmen könnte, sondern ebenso für Berechnungen mit Integerzahlen. Wenn Sie keine reinen Assemblerprogramme schreiben, sondern Assemblerroutinen als Ersatz für besonders zeitkritische Basic-Programmteile verwenden, seien Sie bitte vorsichtig: Wenn die betreffende Routine vom Basic-Programm aufgerufen wird und Werte in die Fließkommaakkumulatoren schreibt, dürfen Sie getrost davon ausgehen, daß nach der Rückkehr ins Basic-Programm und vor dem nächsten Aufruf der Routine diese Werte durch den Basic-Interpreter überschrieben werden. Speichern Sie daher in diesem Bereich bitte keine Werte, die permanent - auch nach der Rückkehr ins Basic-Programm – erhalten bleiben sollen.

Die beschriebenen Bereiche stellen ausreichend Platz für Pointer und oft benötigte Variablen zur Verfügung. Sollten Sie in Ihren Programmen jedoch Tabellen verwalten müssen, oder wollen Sie vor Beginn des Basic-RAMs gar eine komplette kleinere Maschinenroutine ablegen, benötigen Sie größere zusammenhängende Speicherbereiche. Schauen wir uns den Speicher von $FF (Ende Zeropage) aufwärts an.

Größere freie Speicherbereiche

Vor Beginn des Basic-Bereichs bietet der C16 mehrere Speicherbereiche, die für Tabellen und eventuell sogar für kleine Maschinenroutinen genutzt werden können:

1. $0200 bis $02AC: Ein riesiger uns zur Verfügung stehender Bereich, der normalerweise vom Basic-Interpreter und vom DOS als Eingabepuffer verwendet wird (zum Beispiel INPUT #-Befehl).

2. $025D bis $02AC: Diesen Bereich verwendet ebenfalls das DOS. Er darf nur benutzt werden, wenn im Moment des Aufrufs der Maschinenroutine keine (!) Kanäle geöffnet sind. Die vom DOS benötigten Parameter (Filename, Sekundäradresse, Geräteadresse etc.) werden sonst überschrieben.

3. $0332 bis $03F6: Kassettenpuffer und Zähler für Kassettenoperationen. Wenn Sie in diesem Puffer Daten ablegen, bleiben sie erhalten, bis die nächste Kassettenoperation erfolgt. Damit eignet sich dieser Bereich zum Beispiel für kleinere Maschinenroutinen (wenn das Basic-Programm keine Kassettenoperationen durchführt!).

4. $03F7 bis $0436: RS232-Puffer. Wenn Sie die RS232-Schnittstelle nicht verwenden (zum Beispiel für den Betrieb eines Modems), wird dieser Bereich völlig unbenutzt bleiben und ist daher hervorragend für Daten geeignet, die

permanent erhalten bleiben müssen und nicht durch Basic-Interpreter oder Betriebssystem überschrieben werden dürfen.

5. $055F bis $05E6: In diesem Bereich sind die Zeichenketten abgelegt, mit denen die Funktionstasten belegt sind. Sie können ihn für eigene Zwecke benutzen, wenn Ihr Basic-Programm die Funktionstasten nicht benötigt und deren Belegung daher überschrieben werden darf.

Wie Sie sehen, bietet der C 16 genug freie Speicherbereiche für die Ablage von Tabellen, Pointern und kleinen Assemblerroutinen, sowohl in der Zeropage als auch dahinter, obwohl ich mich auf die größten zusammenhängenden Bereiche beschränkt habe.

Wenn Sie kleine Maschinenprogramme in diesen Bereichen ablegen wollen, beschränken Sie sich bitte auf den Kassettenpuffer, den RS232-Puffer und den Funktionstastenspeicher. In letzterem sind Daten vollkommen sicher, in den beiden anderen Bereichen besteht keine Überschreibgefahr, solange weder Kassettenoperationen vorgenommen noch die RS232-Schnittstelle benutzt wird.

Farbspeicher, Bildschirmspeicher, Basic-RAM

Ebenso wie bei vielen meiner Kollegen bestand meine erste Assembler-»Tat« im Beschreiben des Bildschirms mit unsinnigen Zeichen, zum Beispiel Füllen mit lauter Buchstaben. Da ich damals noch eine Scheu vor der Verwendung von Betriebssystemroutinen besaß, wurde jedes Zeichen von mir einzeln in den Bildschirmspeicher »gePOKEt« und die Farbe in der entsprechenden Speicherzelle des Farb-RAMs gesetzt. Für den Fall, daß Sie ebenfalls den Bildschirmspeicher direkt ansprechen wollen, stelle ich Ihnen nun die wichtigsten benötigten Adressen vor:

Farbspeicher: $0800 bis $0BFF
Bildschirmspeicher: $0C00 bis $0FFF

Wichtig sind nur die ersten 1000 Byte des jeweiligen Bereichs (25 Spalten x 40 Zeichen = 1000 Zeichen). Daß beide Bereiche dennoch ein volles Kilobyte umfassen, ist mitunter sehr vorteilhaft, zum Beispiel wenn der gesamte Bildschirm gelöscht oder mit einem bestimmten Zeichen beschrieben werden soll. In diesen Fällen können Sie mit einem recht einfachen Programm volle vier Seiten, das heißt 1024 Byte, beschreiben. Da der relevante Teil des Bildschirm- beziehungsweise Farbspeichers nur 1000 Byte umfaßt, beschreiben Sie zwar mehr Speicherzellen als unbedingt nötig, das Programm wird dadurch jedoch erheblich vereinfacht und kürzer.

Das folgende Programmbeispiel (Listing 1) füllt den Bildschirm des C 16 mit dem Zeichen »A«. Geben Sie es bitte mit dem eingebauten Monitor ein, und starten Sie es anschließend mit »SYS 1015«:

```
. 03f7  a9 00    lda #$00
. 03f9  85 d8    sta $d8
. 03fb  85 da    sta $da
. 03fd  a9 08    lda #$08
. 03ff  85 d9    sta $d9
. 0401  a9 0c    lda #$0c
. 0403  85 db    sta $db
. 0405  a9 01    lda #$01
. 0407  a2 04    ldx #$04
. 0409  a0 00    ldy #$00
. 040b  91 d8    sta ($d8),y
. 040d  91 da    sta ($da),y
. 040f  88       dey
. 0410  d0 f9    bne $040b
. 0412  e6 d9    inc $d9
. 0414  e6 db    inc $db
. 0416  ca       dex
. 0417  d0 f2    bne $040b
. 0419  60       rts
```

Listing 1. »Fill Screen«

Programmablauf: Das Programm liegt im RS232-Puffer und ist damit vor dem Überschreiben geschützt, solange diese Schnittstelle nicht benutzt wird. Da das Programm mit indirekt indizierter Adressierung arbeitet, werden zuerst zwei Pointer angelegt: ein Pointer auf den Beginn des Farb-RAMs ($0800) und ein zweiter Pointer auf den Beginn des Bildschirm-RAMs ($0C00). Der erste Pointer kommt nach $D8/$D9, der zweite Pointer nach $DA/$DB, das heißt in den speziell für Anwendungssoftware reservierten freien Zeropage-Bereich.

Nachdem die Pointer angelegt wurden, werden sowohl Bildschirm- als auch Farbspeicher mit dem Wert eins beschrieben, und zwar jeweils vier Seiten (X-Register wird mit vier geladen). Das Resultat besteht darin, daß der Bildschirm mit dem Zeichen »A« gefüllt wird (1=Bildschirmcode von »A«).

Speichern Sie dieses Programm aus dem Monitor heraus bitte mit »S"FILL SCREEN",8,03F7,041A« auf Diskette beziehungsweise mit »S"FILL SCREEN",1,03F7,041A« auf Kassette ab, da wir es in Kürze wieder verwenden werden.

Einbindung von Maschinenroutinen

Das vorgestellte Programm paßt gerade noch in den RS232-Puffer. Bei größeren Programmen stellt sich jedoch die Frage, wo wir sie im Speicher ablegen können. Für Maschinenprogramme, die ein oder gar mehrere Kilobyte umfassen, steht nur ein zusammenhängender Speicherbereich zur Verfügung, der groß genug ist: der Basic-Speicher selbst.

Der Basic-Speicherbereich beginnt ab $1000. Wo er endet, hängt von mehreren Faktoren ab:

1. Normalerweise endet der Basic-Bereich bei $3FFF.
2. Wurde die hochauflösende Grafik eingeschaltet, wofür zusätzlich Speicherplatz benötigt wird, endet Basic bei $17FF.
3. Wird eine Speichererweiterung verwendet, hängt das Ende des Basic-Speichers von der Größe dieser Erweiterung ab.

Unveränderlich ist somit nur der Beginn des Basic-Speichers. Wenn Sie Wert darauf legen, daß Ihre Maschinenprogramme auch mit Grafik und Speichererweiterungen zusammenarbeiten, benutzen Sie bitte die Zeropage-Adressen $33 und $34, um das Ende des Basic-Bereichs zu ermitteln. Diese Adresse stellt einen Pointer ($33=Low/$34=High) dar, der auf das aktuelle Ende des verfügbaren Speichers zeigt. Wenn keine Erweiterung verwendet wird und die hochauflösende Grafik nicht eingeschaltet wurde, zeigt dieser Pointer auf die Adresse $3FFF (probieren Sie's aus). Verwenden Sie in Ihren Programmen möglichst immer diesen Pointer anstelle einer absoluten Adresse, die sich – abhängig von der Grafik und Speichererweiterungen – ändern kann.

Wenn Sie reine Assemblerprogramme schreiben, zum Beispiel eine Textverarbeitung oder ein Malprogramm, benutzen Sie einfach den Basic-Speicher ab $1000. Häufiger dürfte jedoch die Verwendung von Assemblerroutinen sein, die mit einem Basic-Programm zusammenarbeiten sollen, zum Beispiel Eingabe- oder Sortierroutinen.

In einem solchen Fall legt man üblicherweise die Assemblerroutinen ans Ende des Basic-Speichers und setzt den Zeiger auf dieses Ende ($33/$34) herab, um ein Überschreiben durch Strings zu verhindern, die der Basic-Interpreter leider ebenfalls vom Ende des Basic-Speichers nach unten anlegt.

Im Falle des C 16 ist diese Vorgehensweise etwas problematisch, da das Einschalten der hochauflösenden Grafik genügt, um die Routinen durch den Grafikbildschirm zu überschreiben. Eine flexiblere Methode, die sich mit der Grafik und auch mit eventuellen Speichererweiterungen verträgt, besteht darin, die Routinen vor (!) das Basic-Programm zu legen. Die Adressen $2B/$2C enthalten einen Zeiger auf

den Basic-Anfang. Diesen Zeiger können wir verändern und uns vor dem Basic-Programm beliebig Platz schaffen. Die Vorgehensweise:

1. Zeiger auf Basic-Anfang erhöhen.
2. Null in die neue Startadresse »POKEn«, da am Basic-Anfang immer (!) eine Null stehen muß.
3. »NEW« eingeben, damit der Basic-Interpreter alle weiteren Zeiger entsprechend korrigiert.
4. Maschinenroutinen und Basic-Programm laden (sofern erstere nicht bereits in Form von DATA-Zeilen im Basic-Programm enthalten sind).

Beispiel: 1 KByte vorm Basic-Anfang für Maschinenroutinen reservieren (Listing 2):

```
10 POKE 44,PEEK(44)+4
20 POKE(256*PEEK(44)),0
30 NEW
```

Listing 2. »INT«

In Zeile 10 wird der Basic-Anfang um 1 KByte heraufgesetzt, das heißt in das High-Byte des Pointers auf den Basic-Start ($33/$34 = dezimal 43/44) wird der bisherige Wert plus vier geschrieben. Die Erhöhung des High-Bytes um den Wert vier entspricht vier Seiten, also einem Kilobyte.

Anschließend wird in die Speicherstelle, auf die das High-Byte nun zeigt, eine Null »gePOKEt«, und durch »NEW« werden zuletzt alle übrigen Zeiger korrigiert. Wenn Sie dieses Programm eingeben, starten und anschließend mit »FRE(0)« den freien Speicherplatz ermitteln, werden Sie feststellen, daß ein Kilobyte weniger als gewohnt zur Verfügung steht, jenes Kilobyte, das wir soeben für Maschinenprogramme reserviert haben.

Speichern Sie das Programm bitte unter dem Namen »INIT« auf Diskette oder Kassette ab.

Die beschriebene Vorgehensweise zur Einbindung von großen Assemblerprogrammen in Basic-Programme wollen wir nun an einem Beispiel erproben: Laden Sie bitte »INIT« und starten Sie es (sofern Sie das nicht bereits im vorigen Schritt getan hatten). Gehen Sie in den Monitor und laden Sie »FILL SCREEN«. Verschieben Sie die Routine bitte nun mit:

```
T 03F7 041A 1000
```

Die Routine befindet sich nun in unserem für Maschinenprogramme reservierten Ein-Kilobyte-Bereich, der ab $1000 beginnt. Geben Sie folgendes Programm ein (Listing 3):

```
10 FOR i=1 to 10
20  PRINT CHR$ (147)
30  PRINT "zum aufruf der routine taste druecken"
40  GETKEY A$
50  SYS 4096
60  FOR II=1 TO 1000:NEXT
70 NEXT
```

Listing 3. »INT-Demo«

Wie Sie sehen, ist die Einbindung wunderbar gelungen: Ab $1000 (=dezimal 4096) befindet sich die Maschinenroutine im Speicher, ab $2000 beginnt das Basic-Programm, das die Routine mit »SYS 4096« auf Tastendruck aufruft, woraufhin diese den Bildschirm mit »A« beschreibt und anschließend in das Basic-Programm zurückkehrt.

Die Verwendung der Standardbetriebssystemroutinen hat den Nachteil, daß diese sehr allgemein und flexibel, dadurch jedoch für bestimmte Zwecke manchmal etwas zu langsam sind. In extrem zeitkritischen Routinen kommen Sie nicht umhin, sich Ihre eigenen Routinen – zum Beispiel zur Bildschirmausgabe – zu schreiben, die weniger flexibel, dafür jedoch schneller sind. Sollten Sie jemals in die Verlegenheit kommen, C16-Assemblerprogramme auf den C64 oder den C128 umschreiben zu müssen, werden Sie jedoch dankbar für jede von Ihnen verwendete Betriebssystemroutine sein. Die Standardein-/ausgabe-Routinen sind auf allen drei Computern identisch. Identisch bedeutet nicht, daß die Programme selbst gleich sind, sondern die »Benutzeroberfläche«. Das heißt, daß Routinen die gleichen Auswirkungen haben, die gleichen Parameter übergeben werden müssen und sogar, daß die Einsprungadressen identisch sind.

Letzteres dank der sogenannten »Kernel-Sprungtabelle«, die jeder der genannten drei Computer besitzt. Die Betriebssytemroutinen befinden sich zwar von Computer zu Computer in verschiedenen Speicherbereichen, werden jedoch nie direkt, sondern immer indirekt über diese Sprungtabelle angesprungen. Ein Beispiel:

Die Routine BSOUT gibt ein Zeichen auf dem Bildschirm aus. Aufgerufen wird sie mit einem: JSR $FFD2, und zwar auf jedem (!) Commodore-Computer. An Adresse $FFD2 befindet sich ein Sprungbefehl, zum Beispiel JMP $1000, JMP $2000 oder JMP $3000. An welcher Adresse sich die Routine selbst befindet, ist dank der Sprungtabelle für den Benutzer uninteressant, jedenfalls solange er sich an die »offizielle« Methode hält, Betriebssystemroutinen keinesfalls direkt, sondern immer über die Sprungtabelle aufzurufen.

Die wichtigsten Betriebssystemroutinen

Im folgenden werde ich zu jedem Beispielprogramm die entsprechenden Basic-Programmzeilen vorstellen, um vor allem den Umsteigern, die von Basic her kommen, den Umgang mit den Betriebssystemroutinen zu erleichtern.

GETIN ($FFE4)

Die Routine GETIN (Listing 4) liest ein Zeichen von der Tastatur, wartet jedoch nicht darauf, daß eine Taste gedrückt wird, analog dem Basic-Befehl GET. Der ASCII-Code des Zeichens wird im Akku übergeben. Wurde kein Zeichen gelesen, enthält der Akku den Wert Null. Durch das Testen auf Null kann eine Schleife realisiert werden, die erst nach der Eingabe eines Zeichens verlassen wird.

Basic:

```
10 GET A$:IF A$="" THEN 10
```

oder auch:

```
10 GETKEY A$
```

Assembler:

```
. 03f7  20 e4 ff  jsr $ffe4
. 03fa  f0 fb     beq $03f7
. 03fc  60        rts
```

Listing 4. »GETIN«

Starten Sie diese Routine mit »SYS 1015«. Sie kehren nach dem Betätigen einer beliebigen Taste ins Basic zurück.

Mit GETIN lassen sich außer der Eingabe eines einzigen Zeichens auch komplexere Eingaben durchführen. Alle professionellen Eingaberoutinen, die den Basic-Befehl INPUT aufgrund seiner Nachteile (Bildschirm kann gelöscht werden, während der Eingabe sind keinerlei Eingabeprüfungen möglich und so weiter) ersetzen sollen, verwenden diese Betriebssystemroutine.

BSOUT ($FFD2)

Gibt ein Zeichen, das im Akku übergeben wird, auf dem momentanen Ausgabegerät aus, normalerweise dem Bildschirm (Listing 5). Bei der Bildschirmausgabe ist zu beachten, daß das Zeichen an der momentanen Cursorposition ausgegeben und dieser danach automatisch auf das nächste Zeichen gesetzt wird.

```
Basic:
10 PRINT "A";
20 PRINT "B";

Assembler:
. 03f7  a9 41     lda #$41
. 03f9  20 d2 ff jsr $ffd2
. 03fc  a9 42     lda #$42
. 03fe  20 d2 ff jsr $ffd2
. 0401  60        rts
```

Listing 5. »BSOUT«

Ebenso wie das Basic-Programm gibt auch die Assembler-Routine nach dem Aufruf mit »SYS 1015« die Zeichen »A« und »B« (ASCII-Code: $41/$42) auf dem Bildschirm aus.

Wenn Sie bereits länger in Assembler programmieren, wird für Sie ein kleiner Trick bereits selbstverständlich sein, den ich hier verraten möchte: Da alle Betriebssystemroutinen als Unterprogramme geschrieben sind und mit einem »RTS« als letzten Befehl enden, können Sie Ihre Programme kürzen, indem Sie einen »RTS« einsparen, der nach dem Aufruf einer Betriebssystemroutine folgt, und diese anstatt mit »JSR...« mit »JMP...« aufrufen (Listing 6).

```
. 03f7  a9 41     lda #$41
. 03f9  20 d2 ff jsr $ffd2
. 03fc  a9 42     lda #$42
. 03fe  4c d2 ff jmp $ffd2
```

Listing 6. »BSOUT ohne RTS«

PLOT ($FFF0)

Setzt den Cursor an eine übergebene Bildschirmposition oder übergibt die momentane Cursorposition. Ein Basic-Äquivalent zu dieser Routine existiert nicht. In beiden Fällen wird das X-Register zur Übergabe der Zeile und das Y-Register zur Übergabe der Spalte verwendet:

1. Cursor setzen:
   - Carryflag löschen
   - X und Y laden
   - Routine aufrufen

Cursor auf Zeile 10, Spalte 5 setzen (Listing 7):

```
. 03f7  a2 0a     ldx #$0a
. 03f9  a0 05     ldy #$05
. 03fb  18        clc
. 03fc  20 f0 ff jsr $fff0
. 03ff  60        rts
```

Listing 7. Cursorposition setzen

Rufen Sie die Routine mit »SYS 1015:PRINT"HALLO"« auf. Die Zeichenfolge »HALLO« wird ab Spalte 5 von Zeile 10 ausgegeben werden.

Cursorposition holen (Listing 8):

```
. 03f7  38        sec
. 03f8  20 f0 ff jsr $fff0
. 03fb  00        brk
```

Listing 8. Cursorposition holen

Zeile und Spalte werden in X und Y übergeben. Die entsprechenden Registerinhalte werden Ihnen nach dem BREAK angezeigt. Beachten Sie jedoch, daß der Basic-Interpreter einen Zeilenvorschub ausführt. Wegen dieses Zeilenvorschubs enthält das X-Register die Zeile+1, in der sich der Cursor beim Aufruf befand, und das Y-Register den Spaltenwert Null.

Anwendung der bisher beschriebenen Routinen

Mit den beschriebenen Routinen läßt sich bereits eine Menge anfangen. Ein Beispiel: Sie schreiben ein Malprogramm in Assembler. Der Benutzer will ein Bild, das gerade gemalt wurde, komplett löschen. Da dieser Vorgang unwiderruflich ist, stellen Sie eine sogenannte »Sicherheitsabfrage«. Sie fragen den Benutzer:

»SIND SIE SICHER (J/N) ?«

Die Bildschirmausgabe soll in Zeile 10 und Spalte 15 erscheinen, das heißt etwa in der Mitte des Bildschirms. Außer den Tasten »j« und »n« soll keine weitere zur Beantwortung der Frage zulässig sein, der GET-Befehl ist daher hervorragend zur Lösung des Problems geeignet (Listing 9).

```
Basic:

10 PRINT CHR$(19);:REM CURSOR HOME
20 FOR I=1 TO 10
30   PRINT CHR$(17);:REM CURSOR DOWN
40 NEXT I
50 PRINT TAB(15)"SIND SIE SICHER (J/N) ?"
60 GETKEY A$
70 IF A$="J"THEN GOTO 100:REM LOESCHEN
80 IF A$="N" THEN GOTO 200:REM NICHT LOESCHEN
90 GOTO 60:REM FALSCHE EINGABE

Assembler:

. 03f7  a2 0a     ldx #$0a
. 03f9  a0 0e     ldy #$0e
. 03fb  18        clc
. 03fc  20 f0 ff jsr $fff0

. 03ff  a2 00     ldx #$00
. 0401  bd 1c 04 lda $041c,x
. 0404  20 d2 ff jsr $ffd2
. 0407  e8        inx
. 0408  e0 23     cpx #$23
. 040a  d0 f5     bne $0401

. 040c  20 e4 ff jsr $ffe4
. 040f  c9 4a     cmp #$4a
. 0411  f0 07     beq $041a
. 0413  c9 4e     cmp #$4e
. 0415  f0 04     beq $041b
. 0417  4c 0c 04 jmp $040c
. 041a  60        rts
. 041b  60        rts

)041c 53 49 4e 44 20 53 49 45 20 53 49 43 48 45 52
 20 28 4a 2f 4e 29 20 3f
```

Listing 9. Anwendungsbeispiel

Erläuterung:

Im ersten Programmabschnitt wird der Cursor auf Zeile 10 und Spalte 5 gesetzt. Anschließend wird die Zeichenkette »sind sie sicher (j/n) ?«, die 23 Zeichen enthält (X wird mit 23 verglichen), ab dieser Position auf dem Bildschirm ausgegeben. Die Zeichenkette selbst befindet sich im ASCII-Format am Ende des Programms. Geben Sie sie bitte mit Hilfe des Monitor-Befehls (>) ein, mit dem direkt Bytes verändert werden können.

Der letzte Programmteil besteht aus einer Routine, die ein Zeichen von der Tastatur liest, mit den ASCII-Codes der Zeichen »j« beziehungsweise »n« vergleicht ($4A/$4E) und zum

entsprechenden Programmteil verzweigt. Wurde keines der beiden Zeichen eingegeben, wird die Tastaturabfrage wiederholt.

Ein Tip: in Assemblerprogrammen ist es sehr oft notwendig, Zeichenketten auf dem Bildschirm auszugeben. Anstatt wie im vorgestellten Beispielprogramm die Anzahl der auszugebenden Zeichen zu zählen und das Ausgabeende anhand des Wertes von X zu erkennen, können Sie sich das Programmieren erleichtern, wenn Sie an jede Zeichenkette eine »Endemarke« anfügen, zum Beispiel das Byte $00, und auf diese Endemarke prüfen (Listing 10). Das Abzählen der einzelnen Zeichen einer Zeichenkette wird dadurch überflüssig:

```
        ldx #$00
loop    lda char,x
        beq weiter
        jsr $ffd2
        inx
        jmp loop

weiter  ...
        ...
        ...
        ...
        rts

char    Zeichenkette im ASCII-Format+$00
```

Listing 10. Stringausgabe mit Endemarke

BASIN ($FFCF)

Auch diese Routine (Listing 11) kann zur Eingabe von Zeichen verwendet werden. Bei Verwendung von BASIN erscheint der Cursor auf dem Bildschirm, und ein vom Benutzer eingegebenes Zeichen wird automatisch auf dem Bildschirm ausgegeben, analog dem Basic-Befehl INPUT. Ebenso wie beim INPUT-Befehl ist keine Möglichkeit gegeben, unsinnige Eingaben abzufangen, zum Beispiel das Löschen des Bildschirms. Eingegebene Zeichen werden beim Aufruf von BASIN im Akku übergeben.

```
Basic:

10 INPUT A$

Assembler:

. 03f7  a2 00     ldx #$00
. 03f9  20 cf ff  jsr $ffcf
. 03fc  9d 05 04  sta $0405,x
. 03ff  e8        inx
. 0400  c9 0d     cmp #$0d
. 0402  d0 f5     bne $03f9
. 0404  60        rts
```

Listing 11. »BASIN«

Diese Routine liest solange Zeichen ein und speichert sie ab $0405, bis der Benutzer »RETURN« drückt (ASCII-Code $0D), arbeitet somit analog dem INPUT-Befehl.

Rufen Sie die Routine mit »SYS 1015« auf und geben Sie mehrere Zeichen ein, bevor Sie die Eingabe mit »RETURN« beenden. Sehen Sie sich anschließend mit dem Monitor den Bereich ab $0405 an (m 0405), in dem die Eingabe abgelegt wurde.

Ich würde Ihnen aufgrund der beschriebenen Nachteile von BASIN empfehlen, sich mit Hilfe der GETIN-Routine eine eigene INPUT-Routine in Assembler zu schreiben und auf die Verwendung von BASIN zur Tastaturabfrage in professionellen Programmen möglichst zu verzichten.

Programme laden/speichern

Das Betriebssystem des C16 enthält eine sehr komfortable SAVE-Routine, mit der ein beliebiger Speicherbereich auf Kassette oder Diskette gespeichert werden kann. Die ebenfalls vorhandene LOAD-Routine ermöglicht das Laden eines Programms an eine beliebige Adresse. Diese Routinen können vielseitig verwendet werden, um zum Beispiel ein mit der hochauflösenden Grafik gemaltes Bild, eine Bildschirmmaske und andere Speicherbereiche zu speichern beziehungsweise zu laden. Vor der Benutzung dieser Routinen müssen jedoch die beiden Routinen SETNAM und SETLFS zur Vorbereitung aufgerufen werden.

SETLFS ($FFBA)

Vor Benutzung der SAVE- oder LOAD-Routine müssen mit SETLFS eine logische Filenummer (Akku), die Geräteadresse (X-Register) und eine Sekundäradresse (Y-Register) angegeben werden (Listing 12).

Beispiel: logische Filenummer zwei, Gerät: Floppy, Sekundäradresse vier.

```
lda #$02    ;log. Filenummer
ldx #$08    ;Geräteadresse
ldy #$04    ;Sekundäradresse
jsr $ffba
```

Listing 12. »SETLFS«

SETNAM ($FFBD)

Nachdem mit SETLFS die Fileparameter gesetzt wurden, muß mit der Routine SETNAM der Filename angegeben werden, indem im X- und im Y-Register die Adresse übergeben wird, an der dieser im Speicher abgelegt wurde, und im Akku die Länge des Namens (Listing 13):

1. Akku mit der Länge des Filenamens in Zeichen laden.
2. X-Register mit dem Low-Byte der Adresse laden.
3. Y-Register mit dem High-Byte der Adresse laden.
4. SETNAM aufrufen

Beispiel: Ein Speicherbereich soll unter dem Namen »test« gespeichert werden. Der Name »test« befindet sich (in ASCII-Form) an Adresse $025E, der speziell für Filenamen reserviert ist.

```
lda #$04    ;Länge des Filenamens
ldx #$5e    ;Adresse: Low-Byte
ldy #$02    ;Adresse: High-Byte
jsr $ffbd
```

Listing 13. »SETNAM«

Bei der Speicherung auf Datasette ist es nicht notwendig, dem zu speichernden Programm einen Namen zu geben. Auch in diesem Sonderfall muß (!) jedoch vor Benutzung der LOAD- beziehungsweise SAVE-Routine SETNAM aufgerufen werden. Wenn kein Filename verwendet wird, muß im Akku der Wert Null übergeben werden (Länge des Namens: Null Zeichen). Die Inhalte von X- und Y-Register sind beliebig.

LOAD ($FFD5)

Für das Laden eines Programms spielt die in der Vorbereitungsroutine SETLFS übergebene Sekundäradresse eine wichtige Rolle. Die ersten beiden Bytes jedes Programmfiles enthalten die Adresse, an der sich das Programm beim Speichern befand. Wurde vor dem Aufruf von LOAD die Sekundäradresse eins angegeben, wird das Programm an diese Ursprungsadresse geladen. Wurde eine Null als Sekundäradresse übergeben, so wird der Inhalt von X- und Y-Register beim Aufruf von LOAD als Adresse angesehen, an die das Programm geladen werden soll (X=Low-Byte, Y=High-Byte der Adresse).

Der Inhalt des Akkus entscheidet beim Aufruf von LOAD darüber, ob ein Programm geladen oder ein VERIFY durchgeführt werden soll (0=Laden, 1=Verifizieren).

Beispiele:

1. Programm »test« von Datasette an die Ursprungsadresse laden (Listing 14):

```
lda #$02    ;logische Filenummer
ldx #$01    ;Geräteadresse der Datasette
ldy #$01    ;absolut laden
jsr $ffba   ;Fileparameter setzen

lda #$04    ;Länge des Filenamens
ldx #$5e    ;Adresse low
ldy #$02    ;Adresse high
jsr $ffbd   ;Filename setzen

lda #$00    ;Flag für LOAD
jsr $ffd5   ;LOAD-Routine aufrufen
```

Listing 14. »LOAD« von Kassette

2. Programm »test1« von Diskette an die Adresse $1000 laden (Listing 15).

```
lda #$02    ;logische Filenummer
ldx #$08    ;Geräteadresse der Floppy
ldy #$00    ;an die angegebene Adresse laden
jsr $ffba   ;Fileparameter setzen

lda #$05    ;Länge des Filenamens
ldx #$5e    ;Adresse low
ldy #$02    ;Adresse high
jsr $ffbd   ;Filename setzen

lda #$00    ;Flag für LOAD
ldx #$00    ;Ladeadresse low
ldy #$10    ;Ladeadresse high
jsr $ffd5   ;LOAD-Routine aufrufen
```

Listing 15. »LOAD« von Diskette

In beiden Beispielen wird davon ausgegangen, daß sich der Programmname »test« beziehungsweise »test1« in ASCII-Form ab Adresse $025E im Speicher befindet. Ein weiteres Beispiel zum Laden eines Programms ohne Angabe eines Namens (nur bei der Datasette möglich):

3. Programm ohne Angabe eines Filenamens von Datasette absolut laden (Listing 16).

```
lda #$02    ;logische Filenummer
ldx #$04    ;Geräteadresse der Datasette
ldy #$01    ;absolut laden
jsr $ffba   ;Fileparameter setzen

lda #$00    ;Länge des Filenamens
jsr $ffbd   ;Filename setzen

lda #$00    ;Flag für LOAD
jsr $ffd5   ;LOAD-Routine aufrufen
```

Listing 16. »LOAD« von Kassette ohne Angabe eines Filenamens

SAVE ($FFD8)

Mit Hilfe der SAVE-Routine läßt sich ein beliebiger Speicherbereich auf Kassette oder Diskette speichern. Zuvor müssen wiederum SETLFS und SETNAM als Vorbereitungsroutinen aufgerufen werden. Bei SAVE müssen folgende Parameter übergeben werden:

1. Anfangsadresse des zu speichernden Bereichs. Hierzu muß in der Zeropage ein Zwei-Byte-Zeiger (Low/High) auf diese Adresse abgelegt werden. Bei Aufruf der Routine muß der Akku mit der Adresse geladen werden, an der sich der Zeiger befindet.

2. Endadresse +1 (!) des zu speichernden Bereichs. Diese Adresse wird mit dem X- und dem Y-Register übergeben (X=Adresse low, Y=Adresse high).

Beispiele:

1. $1000 bis $2000 unter dem Namen »test« auf Diskette speichern (Listing 17).

```
lda #$02    ;logische Filenummer
ldx #$08    ;Geräteadresse der Floppy
ldy #$02    ;beliebige Sekundäradresse
jsr $ffba   ;Fileparameter setzen

lda #$04    ;Länge des Filenamens
ldx #$5e    ;Adresse low
ldy #$02    ;Adresse high
jsr $ffbd   ;Filename setzen

lda #$00    ;Anfangsadresse low
sta $d8
lda #$10    ;Anfangsadresse high
sta $d9
lda #$d8    ;Adresse des Zeigers
ldx #$01    ;Endadresse+1 low
ldy #$20    ;Endadresse+1 high
jsr $ffd8   ;SAVE-Routine
```

Listing 17. »SAVE« auf Diskette

2. $1234 bis $3456 unter dem Namen »test1« auf Datasette speichern (Listing 18).

```
lda #$02    ;logische Filenummer
ldx #$01    ;Geräteadresse der Datasette
ldy #$02    ;beliebige Sekundäradresse
jsr $ffba   ;Fileparameter setzen

lda #$05    ;Länge des Filenamens
ldx #$5e    ;Adresse low
ldy #$02    ;Adresse high
jsr $ffbd   ;Filename setzen

lda #$34    ;Anfangsadresse low
sta $d8
lda #$12    ;Anfangsadresse high
sta $d9
lda #$d8    ;Adresse des Zeigers
ldx #$57    ;Endadresse+1 low
ldy #$34    ;Endadresse+1 high
jsr $ffd8   ;SAVE-Routine
```

Listing 18. »SAVE« auf Kassette

Anwendung: hochauflösende Grafik speichern und laden. Als praktische Anwendung der LOAD- und der SAVE-Routine will ich Ihnen nun zeigen, wie Sie den Grafikbildschirm speichern und laden können, der den Speicher von $2000 bis $3FFF belegt. Wir benötigen hierzu ein Basic- und ein Maschinenprogramm. Beide sind in einer Disketten- (Listing 19) und einer Kassettenversion (Listing 20) abgebildet.

Geben Sie bitte zuerst die Maschinenroutine mit Hilfe des Monitors ein. Sie speichert den Grafikbereich unter dem Namen »GRAPHIK« auf Datasette beziehungsweise Diskette und lädt den Grafikbildschirm wieder in diesen Bereich.

```
. 03f7  a9 02     lda #$02
. 03f9  a2 08     ldx #$08
. 03fb  a0 02     ldy #$02
. 03fd  20 ba ff  jsr $ffba
. 0400  a9 07     lda #$07
. 0402  a2 5e     ldx #$5e
. 0404  a0 02     ldy #$02
. 0406  20 bd ff  jsr $ffbd
. 0409  a9 00     lda #$00
. 040b  85 d8     sta $d8
. 040d  a9 20     lda #$20
. 040f  85 d9     sta $d9
. 0411  a9 d8     lda #$d8
. 0413  a2 00     ldx #$00
. 0415  a0 40     ldy #$40
. 0417  20 d8 ff  jsr $ffd8
. 041a  60        rts

. 041b  a9 02     lda #$02
. 041d  a2 08     ldx #$08
. 041f  a0 01     ldy #$01
. 0421  20 ba ff  jsr $ffba
. 0424  a9 07     lda #$07
. 0426  a2 5e     ldx #$5e
. 0428  a0 02     ldy #$02
. 042a  20 bd ff  jsr $ffbd
. 042d  a9 00     lda #$00
. 042f  20 d5 ff  jsr $ffd5
. 0432  60        rts
```

Listing 19. »SAVE LOAD« auf und von Diskette

```
. 03f7  a9 02     lda #$02
. 03f9  a2 01     ldx #$01
. 03fb  a0 02     ldy #$02
. 03fd  20 ba ff  jsr $ffba
. 0400  a9 07     lda #$07
. 0402  a2 5e     ldx #$5e
. 0404  a0 02     ldy #$02
. 0406  20 bd ff  jsr $ffbd
. 0409  a9 00     lda #$00
. 040b  85 d8     sta $d8
. 040d  a9 20     lda #$20
. 040f  85 d9     sta $d9
. 0411  a9 d8     lda #$d8
. 0413  a2 00     ldx #$00
. 0415  a0 40     ldy #$40
. 0417  20 d8 ff  jsr $ffd8
. 041a  60        rts

. 041b  a9 02     lda #$02
. 041d  a2 01     ldx #$01
. 041f  a0 01     ldy #$01
. 0421  20 ba ff  jsr $ffba
. 0424  a9 07     lda #$07
. 0426  a2 5e     ldx #$5e
. 0428  a0 02     ldy #$02
. 042a  20 bd ff  jsr $ffbd
. 042d  a9 00     lda #$00
. 042f  20 d5 ff  jsr $ffd5
. 0432  60        rts
. 0433  00        brk
```

Listing 20. »SAVE LOAD« auf und von Kassette

Der einzige Unterschied zwischen der Disketten- und der Kassettenversion besteht in der unterschiedlichen Geräteangabe bei Aufruf der Routine SETFLS.

Geben Sie nun bitte die Disketten- beziehungsweise Kassettenversion des benötigten Basic-Programms ein (Listing 21, 22).

```
10 A$=GRAPHIK:REM FILENAME IN ASCII-
20 FOR I=1 TO LEN(A$):REM FORM NACH 606
30  POKE 605+I,ASC(MID$(A$,I,1))
40 NEXT
50 :
60 GRAPHIC1,1
70 BOX 1,10,10,200,100
80 BOX 1,50,50,300,190
90 PAINT 1,60,60
100 :
110 SYS 1015:REM GRAFIK SPEICHERN
120 SCNCLR:REM SCREEN LOESCHEN
130 SYS 1015+36:REM GRAFIK LADEN
140 GETKEY Aa$
150 GRAPHICO
```

Listing 21. Grafik aufrufen und Maschinenroutine aufrufen (Diskette)

```
10 A$=GRAPHIK:REM FILENAME IN ASCII-
20 FOR I=1 TO LEN(A$):REM FORM NACH 606
30  POKE 605+I,ASC(MID$(A$,I,1))
40 NEXT
50 :
60 GRAPHIC1,1
70 BOX 1,10,10,200,100
80 BOX 1,50,50,300,190
90 PAINT 1,60,60
100 :
110 CHAR 1,0,24,ZURUECKSPULEN/RECORD+PLAY DRUECKEN
120 SYS 1015:REM GRAFIK SPEICHERN
130 CHAR 1,0,24,ZURUECKSPULEN/TASTE DRUECKEN
140 GETKEY A$
150 SCNCLR:REM SCREEN LOESCHEN
160 CHAR 1,0,24,PLAY DRUECKEN
170 SYS 1015+36:REM GRAFIK LADEN
180 GETKEY A$
190 GRAPHICO
```

Listing 22. Grafik erzeugen und Maschinenroutine aufrufen (Kassette)

Das Basic-Programm besteht aus folgenden Teilen:

1. Filename in ASCII-Form ab 606 (=$025E) ablegen
2. Grafik einschalten
3. Rechtecke malen und ausfüllen
4. SAVE-Routine aufrufen
5. Grafik löschen
6. Gespeicherte Grafik laden

Zusätzlich enthält die Kassettenversion noch Hinweise zur Bedienung der Datasette (Zurückspulen vor dem Speichern, Zurückspulen vor dem Laden, Play beziehungsweise Record+Play drücken).

Sollten Sie eine Floppy besitzen, können Sie beobachten, wie sich die Grafik während des Ladevorgangs langsam aufbaut. Bei Verwendung einer Datasette wird bekanntlich der Bildschirm während Kassettenoperationen »weggeklappt«, so daß Sie die Grafik erst nach dem vollständigen Ladevorgang auf dem Bildschirm sehen. Kassettenbesitzer sollten sich in Geduld üben, da das Speichern beziehungsweise Laden einen jeweils acht Kilobyte großen Speicherbereich umfaßt und daher etwa fünf Minuten dauern wird.

(S. Baloui/ah)

Tabellarische Übersicht

Freie Speicherbereiche:

Zeropage:	$D8	bis	$E8	(Anwendersoftware)
	$61	bis	$70	(Fließkommaakkumulatoren)

Sonstige Bereiche:

$0200	bis	$02AC	(Basic-Eingabepuffer/DOS-Arbeitsbereich)
$025D	bis	$02AC	(aktuelle DOS-Parameter, zum Beispiel Filenamen)
$0332	bis	$03F6	(Kassettenpuffer)
$03F7	bis	$0436	(RS232-Puffer)
$055F	bis	$05E6	(Funktionstastenbelegung)

Speicherbereiche/Zeiger:

Farbspeicher	:$0800 bis $0BFF
Bildschirmspeicher	:$0C00 bis $0FFF
Zeiger Basic-Anfang (Standard $1000)	:$2B/$2C
Zeiger Basic-Ende (Standard $3FFF)	:$33/$34

Betriebssytemroutinen:

Adresse	Label	Funktion	Übergabeparameter
$FFE4	GETIN	Zeichen holen	keine; Zeichenübergabe im Akku
$FFCF	BASIN	Zeichen holen bis RETURN	keine; Zeichenübergabe im Akku
$FFD2	BSOUT	Zeichen ausgeben	Zeichen im Akku übergeben
$FFF0	PLOT	Cursor setzen/ Position holen	Zeile(X)/Spalte(Y)/Carry Clear Carry Set/Positionsübergabe in X/Y
$FFBA	SETLFS	Filepar. setzen	Filenr.(A)/Gerät(X)/Sek.Adresse(Y)
$FFBD	SETNAM	Filename setzen	Länge Name(A)/Adresse Name(X/Y)
$FFD5	LOAD	Programm laden/	Load(0)-Verify(1)- Flag(A)/ Ladeadresse Verify durchführen (X/Y) bei relativem Laden, absolutes Laden bei Sek.-Adresse 1 (SETLFS)
$FFD8	SAVE	Programm speichern	Zeropage-Adresse des Zeigers auf die Anfangsadresse(A)/ Endadresse+1(X/Y)

Reaktionstest

Mit diesem Einzeiler können Sie Ihre Reaktionszeit messen. Um die Länge einer Basic-Zeile nicht zu überschreiten, müssen einige Befehle in abgekürzter Schreibweise eingegeben werden. Nach dem Start des Einzeilers erscheint nach einiger Zeit das Signal »T=«. Nun müssen Sie so schnell wie möglich eine beliebige Taste drücken. Nachdem die Reaktionszeit (in Sekunden) auf dem Bildschirm erschienen ist, beginnt auch gleich der nächste Test. Etwaigem Schummeln wurde vorgebeugt: Beim frühzeitigen Drücken einer Taste wird der Test abgebrochen.

```
2 FORT=0TORND (1) *2500+900:NEXT:GETA$:IFA$=
""THENPRINT"T=";:T=TI:GETKEYA$:PRINT (TI-T) /60:RUN
```

(Hannes Kaltenbach/tr)

RESET ohne Reset-Schalter

Nach langem Herumprobieren wurde im Betriebssystem die Routine gefunden, die einen Reset-Schalter überflüssig macht. Die Routine wird vom Basic aus mit dem SYS-Befehl wie folgt angesprungen:

```
SYS 65529
```

Bei diesem Befehl werden alle Zeiger für das Basic und für das Betriebssystem in den Ausgangszustand zurückgesetzt (Basic-Kaltstart). Alle Maschinenprogramme bleiben jedoch erhalten und können wie gewohnt aufgerufen werden.

(J. Kunz/ah)

Grafik und Sound mit dem C 16

Der beste Computer nützt einem wenig, wenn man seine Fähigkeiten nicht einzusetzen weiß. Wir zeigen, wie man mit dem C16 Grafik und Sound programmiert.

Als Commodore 1984 die beiden Computer C 16/C 116 und Plus 4 auf den Markt brachte, ahnte wohl noch niemand, daß sich der C 16 gegen seinen größeren Bruder durchsetzen würde. Doch es geschah noch mehr: Der Plus 4 verschwand schon nach kurzer Zeit wieder von den Ladentischen. Er war für seine mäßige Leistung einfach zu teuer. Der C 16 hingegen konnte sich bis heute auf dem Markt halten. Dies liegt wohl an seinem geringen Preis: er ist im Moment schon für weit unter 300 Mark erhältlich. Für die Ausführung mit Gummitastatur des C 116 sind sogar zirka 50 Mark weniger zu berappen. Ob sich diese Einsparung jedoch lohnt, ist Ansichtssache. Wer zahlt nicht gerne 50 Mark mehr, um auf einer sehr guten und altbewährten Schreibmaschinentastatur ermüdungsfrei arbeiten zu können?

Wenn im folgenden Artikel vom C 16 die Rede ist, so ist damit natürlich ebenso der C 116 gemeint.

Der C 16 eignet sich besonders für Einsteiger, die mal in die Computerei hineinschnüffeln wollen. Der geringe Preis des Gerätes setzt die Hemmschwelle für Anfänger weit herunter. Woran liegt es nun, daß der C 16 trotz seines geringen Preises nicht ein Renner wie beispielsweise der C 64 wurde?

Das einzige große Manko des C 16 ist sein geringer Speicherplatz. Er stellt dem Benutzer nur ganze 16 KByte zur Verfügung. Schaltet man in den Grafikmodus, dann bleiben sogar nur noch 2 KByte für das Programm. Abhilfen in Form von Speichererweiterungen sind vorhanden, aber nicht unbedingt billig. Von einigen Herstellern (außer Commodore) werden sie inzwischen angeboten.

Ansonsten läßt der C 16 kaum Wünsche offen (zumindest in seiner Preisklasse): Erstmals rüstet Commodore die Computer C 16 und Plus 4 mit dem verbesserten Basic 3.5 aus. Endlose Programme mit PEEK und POKE-Sequenzen gehören der Vergangenheit an. Sound und Grafik können bequem über Basic-Befehle programmiert werden (in diesem Punkt ist er sogar dem C 64 überlegen). In diesem Artikel wollen wir uns vor allem der Grafik- und Sound-Programmierung widmen.

Zur Grafik gehören natürlich auch die Farben. Der C 16 bietet im Gegensatz zum VC 20 oder C 64 erstmals die Möglichkeit, 121 verschiedene Farbstufen darzustellen. Diese setzen sich aus 16 Farben zusammen, von denen 15 in 8 Helligkeitsstufen dargestellt werden können. Für die sechzehnte Farbe (Schwarz) gibt es keine Helligkeitsunterschiede.

Das Beispielprogramm »FARBDEMO« (siehe Listing 1) zeigt alle 121 Farben gleichzeitig am Bildschirm. Zur Farbprogrammierung findet man im Befehlssatz den »COLOR«-Befehl. Ihm sollten drei Parameter folgen:

COLOR Bereich, Farbe, Helligkeit

Die Bereich- und Farb-Zuordnung können Sie den Tabellen 1 und 2 entnehmen. Die Helligkeit kann zwischen 0 (sehr dunkel) und 7 (sehr hell) variiert werden. Ebenso, wie man bei den älteren Commodore-Geräten den jeweiligen Farbwert mittels »PEEK« wieder abfragen konnte, gibt es im neuen Basic 3.5 die Funktionen »RCLR(X)«, um die Farbe des Bildschirmpunktes X auszugeben, und »RLUM(X)«, um die Helligkeit der Farbe zu bestimmen.

Die verschiedenen Grafikmodi

Soviel zu den Farben. Mit den Grafiksymbolen des C 16 lassen sich zwar viele einfache Grafiken (Blockgrafiken) erstellen, aber man stößt sehr schnell an die Grenzen des Machbaren. Um auch komplexere Grafiken zu erstellen, muß in den Grafik-Modus umgeschaltet werden. Man unterscheidet zwei Grafik-Modi: zum einen hochauflösende Grafik, zum anderen Multicolor-Grafik. Im HiRes-Modus können 64000 Punkte einzeln angesprochen werden (siehe Bild 1). Hier können jedoch in jeder Cursorposition (8 x 8 Einzelpunkte) nur zwei Farben dargestellt werden (Zeichen- und Hintergrundfarbe). Bis zu vier Farben lassen sich im Multicolor-Modus darstellen, allerdings auf Kosten der Auflösung: Pro Zeichen können nur noch 4 x 8 Punkte dargestellt werden. Daraus ergibt sich eine Gesamtauflösung von 32000 Grafikpunkten (siehe Bild 2).

```
10 REM **********************************
20 REM *                                *
30 REM *   (C 16)  FARBDEMO    (116)    *
40 REM *           --------             *
50 REM *   CHRISTIAN QUIRIN SPITZNER    *
60 REM *   GRUBERSTR. 53, 8011 POING    *
70 REM *   TELEFON:        08121/81100  *
80 REM *                                *
90 REM **********************************
100 COLOR0,4
110 COLOR4,4
120 PRINT"{CLR,DOWN,BLACK,10SPACE}F A R
B D E M O{CTRL-@}B{DOWN,FLASHON,CTRL-@,L
IG.GREEN}"          ================
140 PRINT" H{2SPACE}S W R Z P G B G O B
G R B H D H{CTRL-@,HOME,RVSON,LIG.RED,CT
RL-@,LIG.GREEN}" E  C E O Y U R L E R R
E O L E U E
160 PRINT" L{2SPACE}H I T A R U A L A A
L S A L N L{CTRL-@}G{RVSON}W{CTRL-@,LIG.
GREEN}" L  W E . N P E U B N U B A U L K
 L
180 PRINT" I{2SPACE}A S . . U N . . G .
G . G B E G{CTRL-@}F{RVSON}V{CTRL-@,LIG.
GREEN}" G  R S . . R . . . E . R . R L L
 R
200 PRINT" K{2SPACE}Z . . . . . . . . .
U . U A B U{CTRL-@,CTRL-O,HOME}R{CTRL-@,
LIG.GREEN}" E . . . . . . . . . . E . E
 U L E
220 PRINT" I{2SPACE}. . . . . . . . . .
N . N . A N{CTRL-@}B{HOME}T{CTRL-@,LIG.G
REEN}" T  . . . . . . . . . . . . . . U
."{CTRL-@}J{HOME}R{CTRL-@,LIG.GREEN,CTRL
-@}={HOME}B{CTRL-@,ORANGE }JR7 B 0 £ B1{
CTRL-@,F6,HOME,CTRL-D,CTRL-A}:{2SPACE,LI
G.GREEN}"STOP"J;{CTRL-@,LEFT,HOME,CTRL-N
,CTRL-A}:{2SPACE,ORANGE }IR1 B 16{CTRL-@
}T{HOME,CTRL-X,CTRL-A}:{2SPACE}R 1,I,J{C
TRL-@}C{HOME}":  PRINT"{SPACE,RVSON,SPAC
E,RVOFF}";
300 :  NEXT I
310 :  PRINT
320 NEXTJ
330 GETKEYA$
©64'er
```

Listing. Dieses Demo-Programm bringt alle 121 Farben des C16 auf den Bildschirm (siehe auch Seite 76)

Der Grund für diese Reduzierung der Auflösung läßt sich so erklären:

Im HiRes-Modus genügt ein Bit, um einen Punkt zu setzen, da die Zeichenfarbe hier entweder ein- oder ausgeschaltet wird. Im Multicolor-Modus muß jedem Bildpunkt aber auch noch eine von vier Farben zugeteilt werden. Das heißt, man muß jedem Punkt einen Wert zwischen Null und Drei zuordnen. Hierfür genügt aber nicht mehr ein Bit. Um vier unterschiedliche Zustände darstellen zu können, benötigt man also 2 Bit ($2^2=4$). Da aber im Multicolor-Modus nicht doppelt so viel Speicherplatz aufgewendet werden kann, muß man für Farbe die halbe Auflösung in Kauf nehmen.

Die beiden Grafik-Modi lassen sich mit dem »GRAPHIC«-Befehl ein- und ausschalten. Dem Befehl sollten zwei Parameter folgen, von denen der erste den Modus angibt (Tabelle 3). Ist der zweite Parameter eine Eins, so wird der Grafik-Bildschirm gelöscht. Folgt kein zweiter Parameter oder ist dieser Null, so bleibt die Grafik erhalten. Wollen Sie im Programm erfahren, in welchem Modus Sie sich befinden, so gibt die Funktion »RGR(X)« den gegenwärtigen Modus aus (Tabelle 3). Befindet man sich schon im Grafikmodus und soll der Bildschirm gelöscht werden, so benützt man den »SCNCLR«-Befehl. Verläßt man den Grafikmodus mit »GRAPHIC 0«, so stehen dem Benutzer trotzdem nur 2 KByte Speicher zur Verfügung, da das Grafikbild bis zum erneuten Einschalten erhalten bleibt. Erst nach dem Befehl »GRAPHIC CLR« werden die restlichen 14 KByte freigegeben. In diesem Fall bleibt auch die Grafik nicht mehr erhalten.

In beiden Grafik-Modi befindet sich der Koordinatensprung (0/0) links oben. Damit zum Beispiel beim Konstruieren eines Graphen nicht jedesmal überlegt werden muß, in welchem Modus man sich gerade befindet, gibt es den »SCALE«-Befehl. Schaltet man die Skalierung mit »SCALE 1« ein, so stehen, unabhängig vom Modus, auf den X- und Y-Achsen jeweils die Werte zwischen 0 und 1023 zur Verfügung. Das heißt aber nicht, daß sich auch die Auflösung auf 1048576 (1024 x 1024) Punkte erhöht hat. Es wird vielmehr mit mehreren benachbarten Adressen jeweils ein- und derselbe Punkt angesprochen. Abgeschaltet wird die Skalierung mit »SCALE 0«.

WERT	BEREICH
0	Bildschirm-Hintergrund
1	Vordergrund (Zeichen, Cursor)
2	Zusatzfarbe 1 (für Multicolor)
3	Zusatzfarbe 2 (für Multicolor)
4	Bildschirmrand

Tabelle 1. Die Zuordnung der Farbparameter im Multicolor-Modus

WERT	Farbe	WERT	Farbe
1	Schwarz	9	Orange
2	Weiß	10	Braun
3	Rot	11	Gelbgrün
4	Cyan	12	Rosa
5	Purpur	13	Blaugrün
6	Grün	14	Hellblau
7	Blau	15	Dunkelblau
8	Gelb	16	Hellgrün

Tabelle 2. 16 Grundfarben lassen sich mit dem C16 darstellen

Genau wie im Textmodus, steht dem Benutzer im Grafik-Modus ein Cursor zur Verfügung. Dieser Pixel-Cursor bleibt, solange nicht gezeichnet wird, unsichtbar. Um ihn an eine beliebige Stelle des Bildschirms zu bewegen, benützt man den »LOCATE«-Befehl:

LOCATE X-Koordinate, Y-Koordinate.

Die X- und Y-Werte sind natürlich abhängig vom jeweiligen Modus beziehungsweise Skalierung. Der Pixel-Cursor bleibt jeweils auf dem zuletzt bearbeiteten Punkt stehen und kann mit der Funktion »RDOT(X)« abgefragt werden. Setzt man für X Null ein, so wird die X-Koordinate angegeben. Die Y-Koordinate kann für X gleich 1 in Erfahrung gebracht werden. Setzt man in die »RDOT«-Funktion die Zahl 2 ein, so wird die aktuelle Farbe des Pixel-Cursors ausgegeben. Ein Beispiel LOCATE 100,150:PRINT ROOT (1) ergibt 150.

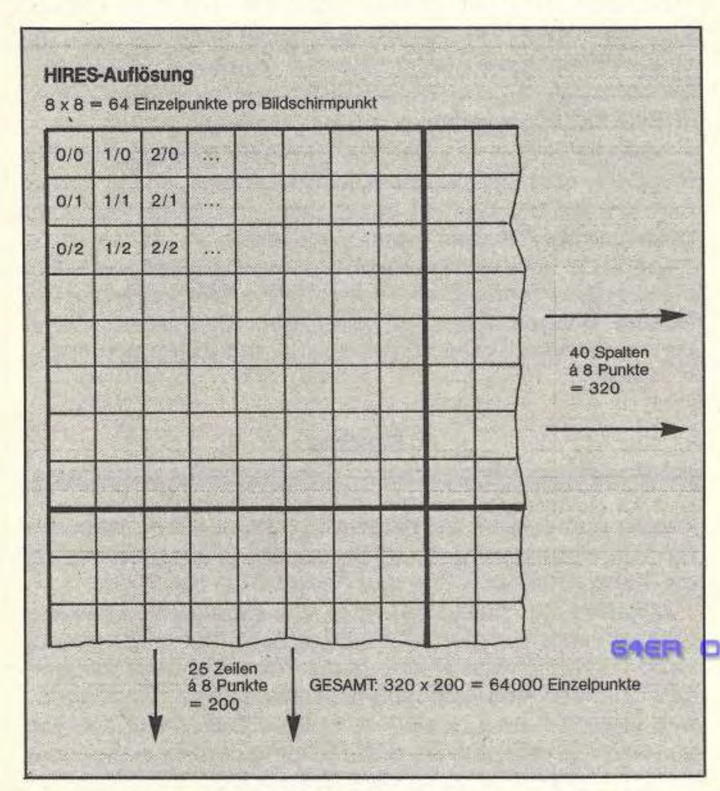


Bild 1. Im HiRes-Modus lassen sich 64000 Punkte darstellen

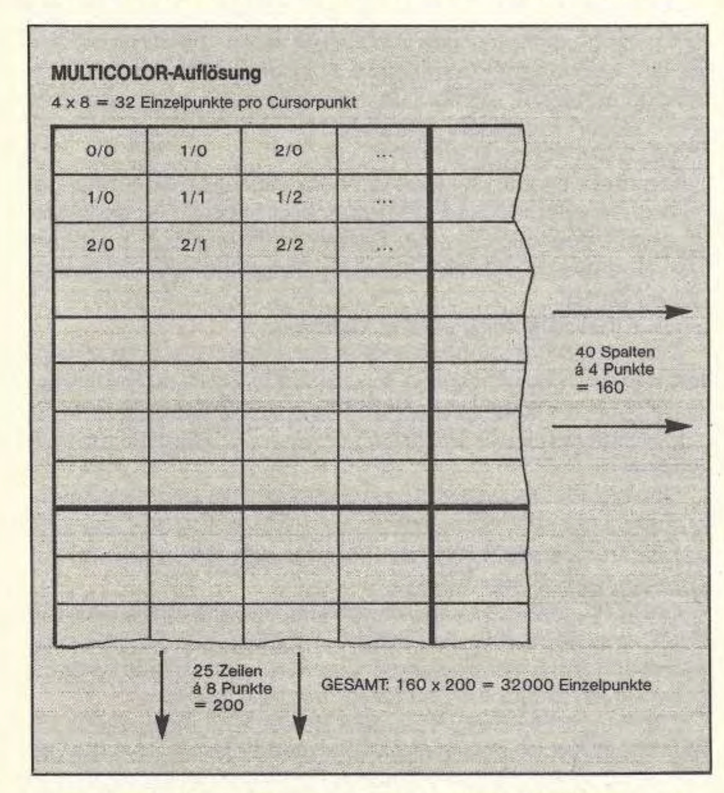


Bild 2. Im Multicolor-Modus reduziert sich die Auflösung auf 32000 Einzelpunkte

Grafikbefehle

Gezeichnet wird mit Hilfe des »DRAW«-Befehls. Mit ihm lassen sich Punkte, Linien und ganze Linienzüge darstellen. Für einen Punkt sind zusätzlich drei Parameter notwendig: Die erste Zahl gibt die Farbe an. Sie kann im HiRes-Modus maximal zwei Werte annehmen:

0 = Hintergrundfarbe (=Punkt löschen)
1 = Zeichenfarbe (=Punkt setzen)

Im Multicolor-Modus stehen zwei zusätzliche Werte zur Auswahl:

2 = Zusatzfarbe 1 (siehe Color-Befehl)
3 = Zusatzfarbe 2 (siehe Color-Befehl)

Der zweite und dritte Parameter des »DRAW«-Befehls geben die X- und Y-Koordinaten des Punktes an. Der Befehl »DRAW 1,160,100« setzt im HiRes-Modus einen Punkt in der Bildschirmmitte. Soll eine Linie gezeichnet werden, so sind weiter zwei Parameter nötig, die die Koordinaten des Endpunktes angeben. Diese beiden Parameter werden mit »TO« abgetrennt:

```
DRAW 1,0,0 TO 319,199
```

Um eine Linienkette zu zeichnen, können weitere Punkte mit »TO« abgetrennt werden.

```
DRAW 1,0,0 TO 319,0 TO 319,199 TO 0,199 TO 0,0
```

Definiert man im »DRAW«-Befehl keinen Anfangspunkt, so wird eine Linie von der letzten Cursorposition zum Endpunkt gezogen.


```
DRAW 1 TO 100,100
```

Mit der »DRAW«-Anweisung lassen sich also auch Rechtecke und Quadrate zeichnen. Allerdings benötigt man die Koordinaten aller Eckpunkte. Rechtwinklige Vierecke lassen sich mit dem »BOX«-Befehl jedoch viel einfacher zeichnen. Hier müssen nur noch Farbe (0 – 3) und die Koordinaten der gegenüberliegenden Eckpunkte angegeben werden. Diese Rechtecke können natürlich auch gedreht dargestellt werden. Zu diesem Zweck gibt man als sechsten Parameter noch den Drehwinkel in Grad ein. Soll das Rechteck zudem noch ausgefüllt werden, muß als siebter Parameter die Zahl 1 folgen. Mit folgendem Beispiel wird in der Mitte des HiRes-Grafikbildschirms ein auf der Spitze stehendes, ausgefülltes Quadrat dargestellt:

```
BOX 1,110,50,210,150,45,1
```

Ein weiterer nützlicher Befehl des Basic 3.5 ist die »CIRCLE«-Anweisung. Mit ihr lassen sich Kreise, Ovale, Kreissegmente und Vielecke zeichnen. Für einen einfachen Kreis genügen schon vier Parameter:

»CIRCLE (Farbzone), (X-Koordinate des Mittelpunkts), (Y-Koordinate des Mittelpunkts), (Radius)«

zum Beispiel: `CIRCLE 1,160,100,100`

WERT	Modus	
0	Textmodus	
1	HiRes-Grafik	
2	HiRes-Grafik	(Zeile 1 - 20)
	& Text	(Zeile 21 - 25)
3	Multicolor-Grafik	
4	Multicolor-Grafik	(Zeile 1 - 20)
	& Text	(Zeile 21 - 25)

Tabelle 3. Fünf verschiedene Grafikmodi stehen dem Programmierer zur Verfügung

An einem Fernsehgerät sieht dieser Kreis eventuell nicht rund, sondern oval aus. Um auch hier einen Kreis zu zeichnen, kann zusätzlich noch eine Zahl für die Höhe eingegeben werden.

```
CIRCLE 1,160,100,100,84
```

Kreissegmente lassen sich mit zwei weiteren Parametern darstellen, indem man noch Anfangs- und Endwinkel (in Grad) angibt. Soll das Ganze auch noch gedreht werden, so kann als achter Parameter noch der Drehwinkel angegeben werden. Aber der »CIRCLE«-Befehl zeichnet nicht nur Kreise, Ovale oder deren Segmente. Es lassen sich ebensogut Dreiecke, Vierecke und Vielecke darstellen. Als Zusatzangabe muß hier als neunter Parameter der Winkel in den Ecken des n-Ecks angegeben werden. Dieser Wert errechnet sich für ein n-Eck folgendermaßen:
Wert = 360/n

Mit der »PAINT«-Anweisung lassen sich geschlossene Flächen farbig ausfüllen. Dem »PAINT«-Befehl folgen maximal vier Parameter: »PAINT (Farbzone), (X-Koordinate), (Y-Koordinate), (Modus)«.

Farbig gefüllt wird die Fläche, in der die angegebenen Koordinaten liegen. Die Grenzen müssen entweder dieselbe Farbe oder (falls Modus 1 gewählt wurde) eine der beiden Zusatzfarben besitzen. Hier ist jedoch jedoch Vorsicht geboten, denn befindet sich irgendwo eine Lücke in der Begrenzung, so wird der gesamte Bildschirm gefüllt.

Schriftzüge

Die schönste Grafik ist ohne Beschriftung aber nur halb so schön. Um Grafiken mit dem nötigen Text zu unterlegen, steht die »CHAR«-Anweisung zur Verfügung. Sie funktioniert ähnlich wie der PRINT-Befehl. Als zusätzliche Parameter sind Farbzone, Zeile, Spalte und eventuell Reverse-Flag gefragt: »CHAR (Farbzone), (Spalte (0-39)), (Zeile (0-24)), (STRING), (REVERS-Flag 0 = AUS; 1 = EIN)«

Folgendes Beispiel schreibt in der fünften Spalte und zehnten Zeile in reverser Schrift »C 16/116«:

```
CHAR 1,4,9,"C 16/116",1
```

Die Fülle dieser Grafik-Befehle ist bestimmt eine Grundlage für kleine oder größere Experimente. Sie werden mit ein wenig Übung und Phantasie feststellen, daß mit geringem Aufwand erstaunliche Effekte erzielt werden können. Im nächsten Absatz werden wir uns mit den Shapes befassen.

SHAPES

Da der C 16 im Gegensatz zum C 64 über keine Sprites verfügt, versucht man sich beim C 16 mit sogenannten »Shapes« zu behelfen. Shapes sind rechtwinklige Ausschnitte aus dem Grafikbildschirm, die einer Stringvariablen übergeben wurden.

Diese Ausschnitte lassen sich dann beliebig an jede Stelle des Bilschirms setzen. Das Shape kann nach Belieben in fünf verschiedenen Modi wiedergegeben werden. Das Shape kann entweder revers oder wie die Originalfläche dargestellt werden. Es läßt sich ebenso mit dem neuen Hintergrund »logisch« verknüpfen. Die entsprechenden Modi können Sie der Tabelle 4 entnehmen.

WERT	Wiedergabe-Modus
0	Wie Original
1	Revers
2	Oder-Verknüpfung mit Grafikfläche
3	UND-Verknüpfung mit Grafikfläche
4	Exklusiv-Oder-Verknüpfung mit Grafikfläche

Tabelle 4. Shapes lassen sich mit Grafiken logisch verknüpfen

Zur Shapehandhabung stehen dem Benutzer zwei Befehle zur Verfügung. Der »SSHAPE«-Befehl übernimmt eine Rechteckfläche aus dem Grafikbild in eine Basic-Stringvariable. Der Befehl hat folgende Syntax:
»SSHAPE Stringvariable, X1,Y1,X2,Y2«

Hierbei geben die vier Parameter die Eckkoordinaten des Shapes an. X2 und Y2 liegen diagonal gegenüber den Koordinaten X1 und Y1. Auch hier sind »scalierte« Werte möglich. Da aber eine Stringvariable nur maximal 255 Zeichen lang sein darf, können die Shapes nicht beliebig groß sein. Die Größe eines Sprites (wie beim C 64) kann jedoch spielend erreicht werden. Um ein Shape an einer anderen Stelle des Grafik-Bildschirms wiederzugeben, steht der »GSHAPE«-Befehl zur Verfügung. Dieser Befehl hat folgendes Format:
»GSHAPE (Stringvariable), X1, Y1 (Modus)

Weil die Größe des Shapes zu diesem Zeitpunkt schon feststeht, sind hier nur die Koordinaten des linken, oberen Eckpunktes des Shapes anzugeben. Der letzte Parameter bestimmt den Wiedergabemodus (Tabelle 4).

Durch die verschiedenartigen Verknüpfungsmöglichkeiten stehen dem Programmierer erstaunliche Fähigkeiten zur Verfügung. Shapes sind zwar kein Ersatz für Sprites, können aber auch zu reizvollen Ergebnissen und Effekten führen.

Sound

Warum Commodore bei diesem Computer auf die altbewährten Sprites verzichtet hat, ist ebensowenig zu verstehen wie der Verzicht auf den SID, den Sound-Chip des C 64.

Während man beim C 64 noch von »Sound«-Programmierung sprechen konnte, begnügt sich der C 16 mit »Tongeneratoren«. Tongenerator 1 erzeugt nur Töne, während Tongenerator 2 auch Geräusche von sich geben kann. Generator 1 wird über Stimme 1 gesteuert. Stimme 2 verwaltet den Tongenerator 2, während die dritte Stimme den Rauschgenerator des zweiten Tongenerators übernimmt.

Da die Tongeneratoren weder Filter, Hüllkurven noch sonstige Effekte beherrschen, ist die Tonerzeugung auf dem C 16 recht einfach: Die Lautstärke kann mit dem »VOL«-Befehl zwischen 0 (Lautstärke aus) und 8 (maximale Lautstärke) geregelt werden. Der Befehl setzt sich aus dem Befehlswort einer Zahl zusammen:
»VOL x (zwischen 0 und 8)«

Die drei Stimmen werden mit dem »SOUND«-Befehl angesprochen. Hier müssen drei Parameter folgen. Der erste gibt Auskunft über die Stimme. Parameter Nummer 2 bestimmt den Notenwert, während der dritte Parameter die Länge des Tones angibt:
»SOUND (Stimme), (Note), (Länge)«

Der Wert für die Tonhöhe kann zwischen 0 und 1015 variiert werden. Die Länge des Tones wird in 1/50stel Sekunden angegeben und kann zwischen 0 und 65535 liegen. Es kann somit eine maximale Tonlänge von über 20 Minuten erreicht werden.

Obwohl die Tongeneratoren mit dem SID des C 64 nicht mithalten können, bieten sich genug Möglichkeiten, sie auszunutzen. Vor allem Stimme 3 eignet sich hervorragend zur Erzeugung schöner Effekte.

Der C 16 zeichnet sich dadurch aus, daß alle Möglichkeiten des Computers ohne die lästigen »PEEKs« und »POKEs« voll ausgeschöpft werden können. Das Basic 3.5 läßt kaum Wünsche offen und ist für Einsteiger bestens geeignet.

Zur Programmierung schneller Spiele wird man aber auch hier kaum die Maschinensprache umgehen können. Es sei hier auf den Artikel »Maschinensprache auf dem C16« in diesem Sonderheft verwiesen.

(C. Q. Spitzner/tr)



Das ist der C16



Bild 1. Das komplette C16-System mit Computer, Datasette und Joystick ist der ideale Einstieg in die Programmierung

Als Nachfolger des VC20 konzipiert und mit einem bemerkenswert komfortablen Basic versehen, ist der C16 inzwischen zum vielgekauften Einsteigermodell von Commodore avanciert – und das gleich doppelt, nämlich als C16 und C116.

Bei dem C16/C116 von Commodore handelt es sich nicht nur um einen Computer, sondern um ein ganzes System (Bild 1, 6 und 7). Computer, Datasette und Joysticks präsentieren sich im schwarz-grauen »Profi-Look«, Drucker und Floppy-Laufwerk des VC20/C64-Systems passen auch an den C16.

Die beiden Geräte C 16 und C 116 sind hard- und softwaremäßig völlig identisch. Der einzige Unterschied besteht bei Gehäuse und Tastatur. Während der C16 hier der durch VC20 und C 64 vorgegebenen Linie folgt, ist der 116 mit seiner Radiergummi-Tastatur und dem Miniatur-Gehäuse eher als Billig-Version einzustufen.

Doch sehen wir uns zunächst den C16 etwas genauer an. Ein erster Blick auf die Tastatur (Bild 2) offenbart schon einige Unterschiede zum VC20/C 64. Aus den beiden Cursortasten des Vorgängers wurden beim C16 deren vier, die aber leider etwas ungünstig rechts oben in einer Reihe in den Tastaturblock integriert wurden. Um für diese Anordnung Platz zu schaffen, mußten einige andere Tasten verlegt werden. So findet der an die VC20-Tastatur gewöhnte Programmierer die häufig benötigten Tasten »+«, »-«, »*«, »↑« und »=« nicht mehr an ihrem gewohnten Platz, was zu Anfang recht lästig ist. Insbesondere die Anordnung der »=«-Taste ganz rechts unten ist sehr unglücklich gewählt.

HELP eingebaut

Die unterste Funktionstaste ist mit »HELP« beschriftet und hat eine spezielle Bedeutung bei der Fehlersuche. Drückt man nämlich diese Taste, nachdem der Computer eine Fehlermeldung angezeigt hat, dann wird die fehlerhafte Zeile sofort aufgelistet und der Abschnitt, in dem der Fehler auftrat, blinkt mit der Cursorfrequenz. Eine feine Sache bei der Programmentwicklung, allerdings wäre es schön, wenn man das mit der Zeit doch etwas nervende Blinken auf irgendeine einfache Art abstellen könnte.

Eine Restore-Taste gibt es nicht mehr, die Linkspfeiltaste des VC20/C64 ist jetzt mit »ESC« (mehr darüber später) beschriftet und im Vordergrund fallen zwei neu beschriftete Tasten, »FLASH ON« und »FLASH OFF«, ins Auge. Zusammen mit der CTRL-Taste wird dadurch ein Blink-Modus einbeziehungsweise ausgeschaltet (analog zu RVS ON/OFF).

Die Funktionstasten sind mit Basic-Befehlen belegt. Sonst entspricht sowohl die Tastatur als auch der Zeichensatz dem »Commodore-Standard«, man wird als »Umsteiger« wenig Schwierigkeiten haben.



Bild 2. Die Tastatur ist bewährt und gut. Erkennbar ist die gegenüber dem VC20 geänderte Belegung einiger Tasten.

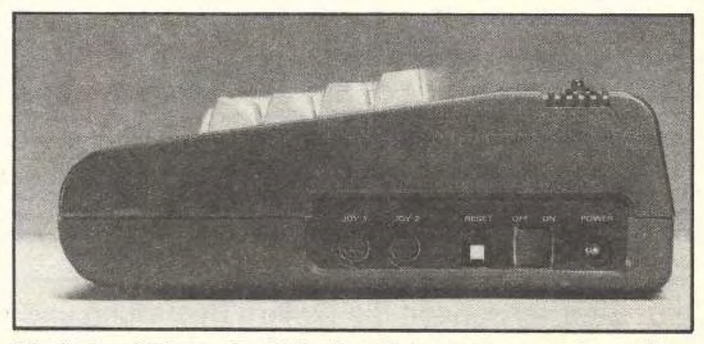


Bild 3. Der C16 von der Seite. Deutlich zu erkennen der weiße Reset-Schalter und die Mikro-Joystickbuchsen. Auch der Netzteil-Anschluß wurde geändert.

Ein Blick an die rechte Seite des C16, wo man ganz richtig den Einschaltknopf vermutet, und sofort fallen zwei Dinge ins Auge (Bild 3). Als erstes und zwar sehr positiv, ein kleiner Reset-Schalter - unverständlich, daß es Computer gibt, die keinen haben. Links daneben zwei winzige Buchsen, darüber steht etwas geschrieben. Man liest es, reibt sich die Augen, schaut nochmals hin - tatsächlich, es ist wahr: Die Mikro-Buchsen sind mit JOY 1 und JOY 2 beschriftet. Endlich einmal ein Computer, an den garantiert kein Joystick außer einem ganz speziellen Commodore-Stick mehr paßt. Diese Joystickanschlüsse sollen über eine verbesserte Abschirmung verfügen, aber es bleibt die Frage, ob man den gleichen Effekt nicht auch mit Standard-Buchsen hätte erreichen können.

Was macht nun jemand, der zum Beispiel vom VC20 auf den C16 umsteigt, mit seinen vorhandenen Joysticks? Nun, vermutlich das gleiche wie mit seiner Datasette, denn auch der Datasettenanschluß wurde geändert. Da hilft alles nichts, entweder wird im Eigenbau ein entsprechender Zwischenstecker hergestellt, oder die alte Datasette wandert zusammen mit dem Lieblingsjoystick in eine Verkaufsanzeige.

Problemlos erweiterbar?

Ein ängstlicher Blick auf die Rückseite des Computers (Bild 4) zeigt, daß wenigstens der serielle Bus nicht mit Spezialbuchsen versehen wurde. Floppy und Drucker sind also weiterhin problemlos anzuschließen. Ein Video-Modulator ist wie beim C64 fest eingebaut. Auffällig ist das Fehlen eines User-Ports, bisher Kennzeichen aller Commodore-Computer.

Der Expansion-Port dient zum Aufnehmen von Steckmodulen mit fertiger Software, sowie zum Anschluß einer (noch nicht erhältlichen) Speichererweiterung. Zu diesem Thema wäre zu bemerken, daß das Betriebssystem mit zwei Speicherbanks arbeitet. Zwischen den 32 KByte ROM von Betriebssystem und Basic und den (noch) 16 KByte RAM wird mittels Bank-Switching hin- und hergeschaltet. Dadurch kann man mit PEEK nicht ins ROM »hineinschauen«, sondern bewegt sich nur auf der RAM-Ebene. Nach Einbau einer 64 KByte-RAM-Erweiterung (siehe Testbericht in diesem Sonderheft) stehen daher tatsächlich fast 60 KByte für Basic-Programme zur Verfügung.

Eine solche RAM-Erweiterung hätte übrigens auch noch bequem im Gehäuse des C16 Platz. Ein Blick dort hinein auf die Platine (Bild 5) offenbart ein sehr aufgeräumtes Innenleben. Die großen integrierten Bausteine, insbesondere die neue CPU 7501 (kompatibel mit 6502/6510) und der ebenfalls neuentwickelte TED 7360, stellen Eigenentwicklungen von Commodore dar und werden nicht frei gehandelt. Informationen über diese Bausteine gibt es daher - ganz nach Art des Hauses - praktisch keine.

Auffällig ist das Fehlen weiterer Peripheriebausteine wie VIA oder CIA. Die Funktionen dieser Bausteine wurden in den TED integriert, der sich auch um die Video-Darstellung und die Tonerzeugung kümmert und so den Prozessor entlastet.

Betriebssystem und Basic sind in je einem 128 KBit ROM untergebracht, die 16 KByte RAM befinden sich in zwei TMS 4416-Chips. Zur besseren Wärmeableitung und Abschirmung ist über der Platine eine gelochte Metallplatte angebracht (im Bild 5 entfernt), von der ein Ausleger direkt mit dem besonders kühlungsbedürftigen TED-Baustein Kontakt hat.

Besonders bemerkenswert: Bei unserem Testgerät waren alle hochintegrierten IC gesockelt. Es bleibt abzuwarten, ob diese gute Technik auch bei größeren Stückzahlen beibehalten wird. Und größere Stückzahlen wurden von diesem Computer bereits verkauft, dafür sorgte aller Inkompatibilität zu anderen Commodore-Computern zum Trotz schon das bemerkenswert gute und umfangreiche Basic 3.5, das den C16 zum idealen Computer für alle diejenigen macht, die wenig mit Maschinensprache im Sinn haben, aber trotzdem gute Programme schreiben wollen.

Programmieren ohne POKEs

Gute Programme für C64 und VC20 zeichnen sich im wesentlichen dadurch aus, daß man sie nicht mehr lesen kann. Nach LIST flimmert dort, zumindest bei Programmen, die mit Grafik und Tonuntermalung arbeiten, ein unergründliches Gemisch von POKE, PEEK, SYS und sehr viel DATA über den Bildschirm, ab und zu auch einmal ein »normaler« Basic-Befehl (in der Regel ein »GOTO«). Wenn man sich einmal vor Augen hält, daß POKE, PEEK und SYS die Verbindung zwischen Basic und Maschinensprache herstellen, dann kann man ruhigen Gewissens sagen, daß die bisherigen Commodore-Heimcomputer im wesentlichen in Maschinensprache programmiert werden mußten - unverständlich für den Einsteiger, schwer erträglich aber auch für den Profi, der

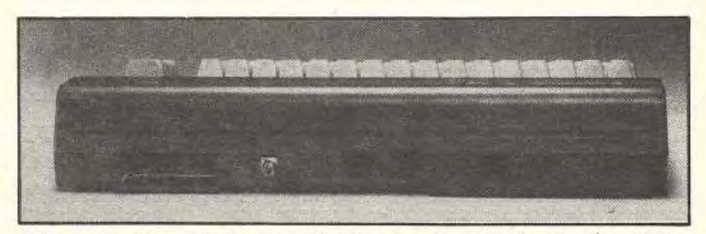


Bild 4. Die Anschlußmöglichkeiten an der Rückseite (von links nach rechts): Expansion-Port, Antennenbuchse, Monitorausgang, serieller Port und Datasetten-Anschluß.

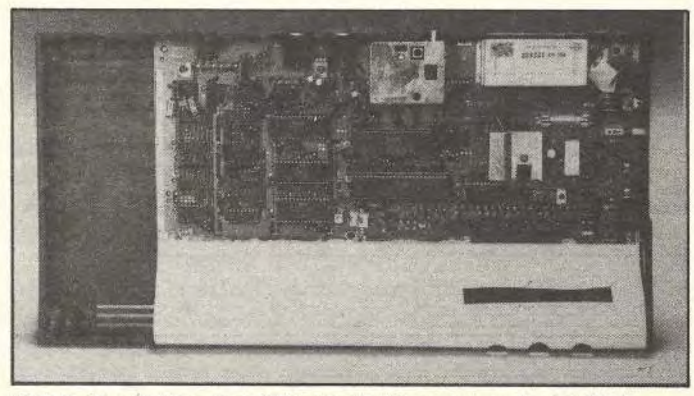


Bild 5. Die Platine des C16. Im Gehäuse ist noch viel Platz.



Bild 6. Die Datasette präsentiert sich auch im neuen Design

kostbare Programmierzeit damit vergeudete, sich seine eigene Basic-Erweiterung zu basteln, um überhaupt erst vernünftig arbeiten zu können.

Mit dem Basic 3.5 hat Commodore ganz offensichtlich einen anderen Weg beschritten. Von der simplen Farbwahl über die Tonerzeugung bis hin zur hochauflösenden Grafik läßt sich alles mit entsprechend leistungsfähigen Basic-Befehlen programmieren. Daneben wird natürlich auch die strukturierte Programmierung unterstützt. Sprachkonstruktionen wie IF...THEN...ELSE, DO WHILE oder DO UNTIL machen die dem Programm zugrunde liegende Idee im Listing sichtbar und vermeiden umständliche (und langsame) GOTO-Sprünge.

Natürlich ist Basic 3.5 vollständig aufwärtskompatibel zum altertümlichen V.2.0-Minimal-Basic des VC20/C64. Insgesamt ist das Basic 3.5 so leistungsfähig, daß alleine eine genaue Beschreibung aller Befehle und Funktionen leicht ein ganzes Sonderheft füllen würde (Tabelle 1). Beschränken wir uns daher auf die Betrachtung einiger wichtiger Aspekte.

Die Grafik

Der Bildschirm des C16 hat eine Aufteilung von 25 Zeilen zu je 40 Zeichen. Jedes Zeichen wird in einer 8x8-Matrix dargestellt. Damit gibt es insgesamt also 320 Punktpositionen (40x8) pro Zeile, und 200 Punktpositionen (25x8) in vertikaler Richtung. Genausoviele Punkte, nämlich 320 mal 200, können im hochauflösenden Grafik-Modus einzeln angesprochen werden. Mit insgesamt 64000 Einzelpunkten erreicht der C16 damit die Grafik-Auflösung des C64, was in etwa einer Verdopplung gegenüber dem VC20 gleichkommt.

Daneben ist noch ein Mehrfarbenmodus mit halbierter Auflösung vorgesehen, bei dem jeder der dann 32000 Einzelpunkte eine von vier möglichen Farben annehmen kann.

Bis hier herrscht noch eine völlige Übereinstimmung zum Grafik-Konzept des C64. Der große Unterschied ist nun, daß die C16-Grafik vom Basic voll unterstützt wird.

Um beispielsweise mit dem C64 (ohne Erweiterung!) einen Kreis oder eine Ellipse in hochauflösender Grafik zu zeichnen und zusätzlich noch einen Text einzublenden, benötigt man neben fundierten Kenntnissen über die interne Organisation seines Computers und einigen Jahren Programmiererfahrung mindestens einige Dutzend Basic-Zeilen und jede Menge Geduld - denn Basic-Grafik ist langsam, wenn die primitivsten Dinge mühselig simuliert werden müssen. So werden ganze Kurse und Bücher damit gefüllt, auf dem C64 das zu erreichen, wofür man beim C16 drei Basic-Befehle und – als absoluter Neuling – maximal einige Minuten blättern im Handbuch braucht:

```
10 GRAPHIC 1,1 : CIRCLE, 160,100,60,30 :
CHAR,18,12, »HALLO«
```

Mit GRAPHIC 1,1 wird in den hochauflösenden Grafikmodus geschaltet. Der erste Parameter gibt den gewählten Modus an (0 für Text, 1 für Hochauflösung, 2 für Hochauflösung mit Textfenster, 3 für Mehrfarben-Modus, 4 für Mehrfarben-Modus mit Textfenster). Als Textfenster sind dabei die untersten fünf Bildschirmzeilen vorgesehen. Dieses Textfenster ist dann mit normalen PRINT-Befehlen ansprechbar. Will man Text direkt in die hochauflösende Grafik hineinmischen, dann bedient man sich des CHAR-Befehls. Der erste Parameter bestimmt im Mehrfarbenmodus die Textfarbe (und wird daher in unserem Beispiel für hochauflösende Grafik durch ein einzelnes Komma ersetzt). Die beiden folgenden Parameter kennzeichnen die Cursorposition, an welcher der Text ausgegeben werden soll. Als letztes muß noch ein Textstring angegeben werden, der dann ab der spezifizierten Position in die Grafik eingeblendet wird. Ein fünfter Parameter ist optional, nämlich die Angabe, ob der Text normal oder revers erscheinen soll.

Bleibt nur noch der CIRCLE-Befehl, der mit bis zu neun Parametern sehr komplex sein kann. Der erste Parameter gibt wieder die Farbzone an und ist nur im Mehrfarbenmodus zu verwenden. Dann folgen die Mittelpunktkoordinaten und der Radius in X- und Y-Richtung. Die letzten beiden Werte sind bei einem Kreis natürlich gleich groß und daher reicht es, nur den ersten davon anzugeben. In unserem kleinen Beispiel haben wir jedoch eine Ellipse gezeichnet, und zwar mit Mittelpunkt in (160,100) und den Halbachsenabmessungen 60 beziehungsweise 30 Punkte.

Aber der CIRCLE-Befehl ist noch weitaus leistungsfähiger. Weitere Parameter regeln das Zeichnen nur einzelner Segmente sowie eine Drehung der ganzen Figur um einen beliebigen Winkel. Neben dem Zeichnen von Kreisen und Ellipsen kann der CIRCLE-Befehl auch für beliebige Vielecke verwendet werden.

Daneben steht noch eine Anzahl weiterer leistungsfähiger Grafik-Befehle zur Verfügung. DRAW zeichnet Einzelpunkte oder Linien, BOX zaubert blitzschnell alle möglichen Rechtecke auf den Bildschirm. LOCATE dient zum Positionieren des Grafikcursors, mit PAINT werden geschlossene Flächen ausgefüllt. SCALE schließlich dient zur Skalierung der Zeichenfläche und SCNCLR löscht den Bildschirm unabhängig vom eingestellten Grafikmodus.



Bild 7. Der Joystick zum C16



Bild 8. Die Gummitasten-Zwilling C16

Shapes statt Sprites

Im Gegensatz zum C64 sind beim C16 keine Sprites vorgesehen. Dafür gibt es jedoch leistungsstarke Basic-Befehle, um Bildausschnitte aus der hochauflösenden Grafik – sogenannte »Shapes« – in Stringvariable abzuspeichern oder wieder auf den Bildschirm zu bringen. Zum Beispiel wird mit »SSHAPE A$,100,100,120,130« der Inhalt des Rechtecks mit linker oberer Ecke (100,100) und rechter unterer Ecke (120,130) auf dem Grafikbildschirm in der Stringvariablen A$ abgelegt. Mit »GSHAPE A$, 60,70« wird die Grafikinformation aus A$ wieder als Rechteck mit linker oberer Ecke (60,70) abgelegt. Über einen zusätzlichen (optionalen) Parameter kann der Wiedergabemodus bestimmt werden: Shapes können genauso wie aufgenommen auch wieder eingeblendet werden (und überschreiben dabei den Hintergrund), sie können revers dargestellt und schließlich wahlweise auch ODER, UND oder EXKLUSIV-ODER mit dem Hintergrund verknüpft werden.

Der C16 hat eine Grundpalette von 16 Farben zur Verfügung, von denen jede (bis auf Schwarz) noch in acht verschiedenen Intensitätsstufen dargestellt werden kann. Das ergibt insgesamt eine beachtliche Auswahl von 121 Farbtönen.

Mit dem COLOR-Kommando können dabei die Farben für Bildschirmrahmen, Hintergrund, Mehrfarbenmodus und auch die Zeichenfarbe gewählt werden.

Zur Tonerzeugung stehen zwei unabhängige Tongeneratoren zur Verfügung, von denen einer auch für Geräuscheffekte eingesetzt werden kann. POKE-Befehle sind auch hier nicht nötig. Mit dem SOUND-Befehl werden sowohl der gewünschte Tongenerator angewählt als auch Tonhöhe (Notenwert) und Klangdauer angegeben, mit VOL wird die Lautstärke eingestellt.

Das Besondere dabei ist, daß bis zu zwei SOUND-Befehle (einer pro Kanal) parallel zum Basic-Programm ausgeführt werden. Der Programmablauf wird durch einen SOUND-Befehl also nicht etwa aufgehalten, bis der Ton zu Ende gespielt wurde; das Programm läuft normal weiter, während der Ton entsprechend der gewählten Tondauer erklingt. Man kann sich wohl vorstellen, wie diese Fähigkeit ein Programm beschleunigt, wenn man sich vor Augen hält, daß Computer der Vorgängergeneration (VC20/C64) die Tondauer über leere FOR...NEXT-Schleifen bestimmten.

Komfortable Programmierung

Bei der Entwicklung von eigenen Programmen und der in der Regel notwendigen Fehlersuche kommen die eingebauten Programmierhilfen des 3.5 Basic erst richtig zur Geltung. Eine Reihe von Befehlen und speziellen Funktionen stehen zur Verfügung, die bisher bei Commodore-Computern unter der Abkürzung OGNV (oft gebraucht, nie vorhanden) liefen.

Eine automatische Zeilennumerierung mittels AUTO ist ebenso selbstverständlich wie ein RENUMBER-Befehl zum Neunumerieren des Programms (wobei wahlweise auch nur Programmteile numeriert werden können und der Zeilenabstand sowie die Startzeile natürlich frei wählbar sind).

Diskettenkommandos, die man früher umständlich mit »OPEN 1,8,15...« an die Floppy senden mußte, sind jetzt als Basic-Kommandos integriert. SCRATCH löscht beispielsweise ein File auf der Diskette. Das Laden und Speichern von Diskettenprogrammen geschieht jetzt mit DLOAD beziehungsweise DSAVE. DIRECTORY holt das Inhaltsverzeichnis der Diskette auf den Bildschirm, selbstverständlich ohne Programmverlust.

Für das häufig benötigte Warten auf einen Tastendruck gibt es den Spezialbefehl GETKEY A$, wodurch man sich das lästige »IF A$="" THEN...« spart.

Hinter RESTORE kann eine Zeilennummer angegeben werden, was das einfache Hantieren mit mehreren unabhängigen DATA-Blöcken erlaubt.

Formatierte Ausgabe ist mit PRINT USING möglich. Die dabei verwendeten Zeichen lassen sich mit dem PUDEF-Kommando umdefinieren. Beispielsweise kann man für die formatierte Ausgabe den (amerikanischen) Dezimalpunkt durch das in Europa übliche Komma ersetzen. Mit ZONE kann die Weite der TAB-Bereiche geändert werden.

Die häufig gebrauchte INSTR-Funktion (hat als Ergebnis die Position eines Teilstrings in einem anderen String) ist ebenso vorhanden wie die Funktion JOY zur einfachen Joystickabfrage.

Bemerkenswert ist auch, daß die MID$-Funktion jetzt auch auf der linken Seite einer Wertzuweisung stehen kann. Sei beispielsweise A$="HALLO". Nach Ausführung des Befehls »MID$(A$,2,1) = "E"« ist A$ dann gleich der Zeichenfolge »HELLO«.

Für die Umrechnung zwischen Dezimal und Hexadezimal sind die beiden Funktionen DEC und HEX$ vorhanden.

Strukturierte Programmierung

Basic 3.5-Programme sind in der Regel um einiges übersichtlicher (und dabei schneller) als VC20/C64-Programme. Der Grund ist einleuchtend: Durch zusätzliche Schleifenbefehle werden GOTO-Anweisungen eingespart und damit entfällt auch die Suchzeit, um die Zeilennummer zu finden.

Daneben wurde auch die IF-Anweisung um die ELSE-Klausel erweitert, was die Programmierung in vielen Fällen vereinfacht.

abs	el	log	rnd
and	end	loop	run
asc	er	mid$	save
atn	err$	monitor	scale
auto	exp	new	scnclr
backup	fn	next	scratch
box	for	not	sgn
char	fre	on	sin
chr$	get	open	sound
circle	getkey	or	spc(
close	get#	paint	sqr
clr	gosub	peek	sshape
cmd	goto	poke	st
collect	graphic	pos	stop
color	gshape	print	str$
cont	header	print#	sys
copy	hex$	printusing	tab(
cos	if	pudef	tan
data	input	rclr	ti
dec	input#	rdot	ti$
def	instr	read	trap
delete	int	rem	troff
dim	joy	rename	tron
directory	key	renumber	until
dload	left$	restore	usr
do	len	resume	val
draw	let	return	verify
ds	list	rgr	vol
ds$	load	right$	wait
dsave	locate	rlum	while

Tabelle 1. Der leistungsfähige Befehlssatz des C16/116. Ein Großteil der Befehle ist beim VC20/C64 nicht vorhanden.

Der Kern der neuen Schleifenstruktur besteht aus den Anweisungen DO und LOOP. Ähnlich wie FOR...NEXT umklammert DO...LOOP einen Programmteil. Die Wirkung ist die folgende: Bei Erreichen eines DO merkt sich der Basic-Interpreter die Adresse dieses DO-Befehls als Schleifenanfang. Wird dann im weiteren Verlauf das zugehörige LOOP gefunden, erfolgt sofort ein Rücksprung zur Position des DO-Befehls. Das ergibt eine »unendliche« Schleife, was allerdings in den meisten Fällen nicht erwünscht ist. Daher ist die EXIT-Anweisung vorgesehen, die ein Verlassen der Schleife und eine Fortsetzung des normalen Programmablaufs hinter dem LOOP-Befehl ermöglicht. In der Regel wird man das EXIT von einer bestimmten Bedingung abhängig machen. Beispiel:

```
10 DO
20 GET A$ : IF A$="X" THEN EXIT
30 LOOP
```

Dieses Programm wartet, bis die Taste X gedrückt wird.

Die (unbedingte) DO...LOOP-Schleife kann unter Verwendung von UNTIL oder WHILE in eine bedingte Schleife abgewandelt werden. DO WHILE ⟨Bedingung⟩ ... LOOP wird ausgeführt, solange die ⟨Bedingung⟩ erfüllt ist. Durch UNTIL wird praktisch der umgekehrte Fall erzeugt. DO UNTIL ⟨Bedingung⟩ ... LOOP wird solange durchlaufen, bis die Bedingung erfüllt ist. Natürlich können auch bei bedingten Schleifen zusätzliche EXITs eingebaut werden. Das ermöglicht sehr effiziente Programme, insbesondere, wenn mehrere Bedingungen beachtet werden müssen.

Gemäß der Parole »wer viel programmiert macht viele Fehler« ist jedes Basic nur so gut wie seine Hilfen zur Fehlersuche und Fehlerbehandlung. Und hier hat der C 16 einiges zu bieten.

Die HELP-Funktion wurde bereits anfangs erwähnt und ermöglicht die schnelle Lokalisierung eines Fehlers innerhalb einer Programmzeile.

Für den nicht seltenen Fall, daß keine Fehlermeldung erfolgt, das Programm jedoch unsinnige Sachen macht (also irgendwo noch ein logischer Fehler steckt) kann man mit TRON eine Trace-Funktion einschalten. Dabei wird die Zeilennummer der gerade abgearbeiteten Zeile angezeigt, wodurch man so manchem Fehler leichter auf die Spur kommen kann. TROFF schalten den Trace wieder ab.

Debugging leicht gemacht

Für die Fehlerbehandlung innerhalb des Programms ist der TRAP-Befehl vorgesehen. Zum Beispiel wird mit der Anweisung »TRAP 1000« beim Auftreten eines Fehlers das Programm nicht mit entsprechender Meldung abgebrochen, sondern es wird in eine Fehlerbehandlungsroutine gesprungen (hier ab Zeile 1000). Die Nummer der Zeile, in der der Fehler auftrat, wird dabei in der Systemvariablen EL gespeichert. Die Variable ER enthält die Fehlernummer, und ERR$ die Fehlermeldung im Klartext. Mit diesen Informationen kann man in der Fehlerbehandlungsroutine entsprechende Maßnahmen ergreifen und schließlich mit RESUME den normalen Programmablauf wieder aufnehmen lassen. Übrigens wird auch das Drücken der Stop-Taste mit TRAP abgefangen.

Window-Technik

Bei soviel Licht fällt gelegentlich auch ein Schatten. Die für den C 16/116 angekündigte moderne Window-Technik, also das Arbeiten mit verschiedenen Bildschirmfenstern, ist leider nicht in einer vollends überzeugenden Form implementiert.

⟨ESC⟩ & Taste	Funktion
A	Automatisch einfügen
B	(Set Bottom) Fixiert an der gegenwärtigen CURSOR-Position die rechte, untere Fensterecke
C	(Clear auto insert) Hebt automatisch Einfügen auf
D	(Delete) Löscht eine Zeile an der CURSOR-Position
I	(Insert) Fügt eine Zeile an der CURSOR-Position ein
J	CURSOR wird an den Anfang der CURSOR-Positionszeile gesetzt
K	CURSOR wird an das Ende der CURSOR-Positionszeile gesetzt
L	Schaltet SCROLLING-Modus ein
M	Schaltet SCROLLING-Modus aus
N	Schaltet zur normalen Bildschirmgröße zurück und löscht den Bildschirm
O	(Off) Hebt Einfüge-, Anführungszeichen-, Reverse- und Blink-Modus wieder auf
P	Löscht Bildschirmzeile vom Anfang bis zur CURSOR-Position
Q	Löscht Bildschirmzeile ab der CURSOR-Position bis zum Ende
R	Verkleinert das Bildschirmformat und löscht den Bildschirm
T	(Set Top) Fixiert an der gegenwärtigen CURSOR-Position die linke, obere Ecke des Fensters
V	SCROLLEN des Bildschirminhalts nach oben
W	SCROLLEN des Bildschirminhalts nach unten
X	(Exit ESC) Befreit Sie aus dem ESCAPE-Modus nach versehentlicher Betätigung der ESC-Taste

Tabelle 2. Die ESC-Funktionen

A	Assemble	Wandelt ein Mnemonic (Klartext-Befehl) in den entsprechenden Maschinencode des 6502 beziehungsweise 7501
C	Compare	Vergleicht zwei Speicherbereiche und zeigt die Unterschiede
D	Disassemble	Wandelt Maschinencode in Mnemonics (Klartext)
F	Fill	Füllt einen Speicherbereich mit wählbarem Wert
G	Go	Startet Maschinenprogramm an angegebener Adresse
H	Hunt	Durchsucht einen Speicherbereich nach einem bestimmten Wert und zeigt alle Speicherplätze an, die diesen Wert enthalten
L	Load	Lädt ein Programm von Kassette oder Diskette
M	Memory	Zeigt alle Inhalte eines wählbaren Speicherbereichs in Hexdarstellung an
R	Register	Ausgabe der aktuellen Registerinhalte
S	Save	Speichert ein Programm auf Kassette oder Diskette
T	Transfer	Blockkopierbefehl, kopiert einen bestimmten Speicherbereich in einen anderen
V	Verify	Vergleicht ein Programm im Arbeitsspeicher mit einem auf Kassette oder Diskette
X	Exit	Zurück zu Basic
.	(Punkt)	Entspricht dem A (Assemble)
>	(größer als)	Ändert bis zu 8 Byte ab bestimmter Speicherstelle (nach M-Befehl)
;	(Semikolon)	Ändert die 7501-Registerinhalte (nach R-Befehl)

Tabelle 3. Die TEDMON-Befehle

Es kann überhaupt nur ein einziges Window erzeugt werden und das nicht etwa per Basic-Befehl (wie man es an sich erwarten würde), sondern über eine ESC-Funktion. Damit kommen wir gleich zur Bedeutung der ESC-Taste auf der Tastatur.

Es gibt nämlich 26 ESC-Funktionen, die durch Drücken von ESC, gefolgt von einer Buchstabentaste, aufgerufen werden (Tabelle 2).

Um das eine mögliche Fenster zu erzeugen, muß man den Cursor in die linke obere Ecke des vorgesehenen Windows bringen, dann ESC T drücken, anschließend in die rechte untere Ecke fahren und ESC B betätigen. Dadurch ist das Fenster definiert. Alle Ein- oder Ausgaben spielen sich jetzt ausschließlich hier ab.

Durch zweimaliges Drücken der Home-Taste wird das Window wieder gelöscht.

So fortgeschritten das Konzept auch gegenüber dem VC20/C64 ist, es bleibt einiges zu wünschen übrig. Die Methode der Window-Definition ist viel zu umständlich und zu langsam, zumal ein Befehl zur direkten Cursorpositionierung nicht vorhanden ist. Das fällt um so schwerer ins Gewicht, als immer nur ein einziges Window definiert werden kann, was aber in der Regel nicht sehr sinnvoll ist. Wenn man den Bildschirm aber in verschiedene Bereiche aufteilen will, dann wirkt sich das ständige umständliche Definieren der Fenster doch zum einen auf die Programmlänge, zum anderen auf die Abarbeitungsgeschwindigkeit negativ aus.

Dennoch ist das Windowing ein Schritt in die richtige Richtung, hin zum benutzerfreundlichen Computer. Für eine übersichtliche Bildschirmaufteilung besteht jedenfalls in fast jeder Programmiersituation ein Bedarf. Wo man sich früher damit behalf, den gesamten Bildschirm bei jeder Veränderung neu aufzubauen, ist es jetzt möglich, nur den wirklich zu ändernden Bereich anzusprechen, ohne dabei die übrigen Informationen zu beeinflussen.

Maschinensprache-Monitor eingebaut

Für Maschinensprachefreunde – und solche, die es werden wollen – hält der C16 noch einen ganz besonderen Leckerbissen parat. Er verfügt nämlich über einen fest im ROM vorhandenen Maschinensprache-Monitor, genannt TEDMON.

TEDMON ist genau genommen sogar mehr als nur ein Monitorprogramm für Maschinensprache. Er enthält nämlich einen Disassembler und auch einen kleinen Assembler. Maschinenprogramme können mit TEDMON sehr komfortabel entwickelt und anschließend als schnelle Unterroutinen von Basic aus aufgerufen werden. Tabelle 3 zeigt den TEDMON-Befehlssatz.

Ein Schwachpunkt des C16 ist sicher der mit 16 KByte zu kleine Anwenderspeicher (nach Einschalten der hochauflösenden Grafik bleiben noch exakt 2045 Byte zum Programmieren). Man sollte aber nicht vergessen, daß man bei der Leistungsstärke des C16 Basic in 2 KByte etwa das gleiche an Grafik-Programm unterbringen kann wie beim C64 in 8 KByte.

Der C16 ist insbesondere vom Basic her in der Tat mindestens eine ganze Generation weiter als der VC20 und der C64. Durch sehr komfortable Programmierhilfen und ein umfangreiches, praxisnahes Basic ist er der ideale Einsteiger-Computer mit der Fähigkeit, seinem Besitzer das Tor zum Computer zu öffnen. (ev)

Den C16 und VC 20 durchschaut

Der VC 20 und der C16 besitzen – so erstaunlich das auch klingen mag – sehr viele Gemeinsamkeiten. Im Rahmen dieses Kurses bekommen Sie einen Einblick in den inneren Aufbau und die Programmierung beider Computer.

Dieser Kurs wird »zweigleisig« abgehandelt. Die im normalen Text angeführten Adressen beziehen sich immer auf das Betriebssystem des VC 20. Die entsprechenden Werte für den C16 sind in Klammern dahinter angegeben. Oft wird es auch vorkommen, daß nur eine Adresse genannt wird, da diese bei beiden Geräten gleich ist.

Die Betriebssysteme des VC 20 und des C16 sind sehr flexibel gestaltet. Es gibt für den Benutzer viele Möglichkeiten, das Bestehende zu ändern oder zu ergänzen.

Wie Basic den Speicher verwaltet

Beginnen wollen wir mit der Organisation des verfügbaren RAM durch den Basic-Interpreter.

Der Basic-Beginn liegt bei Adresse 4096, das Ende bei Adresse 7680 (16383) (die Werte beziehen sich auf die Grundversion). Unmittelbar ab dem Basic-Beginn wird das eigentliche Programm abgelegt. An dessen Ende beginnen die Variablen und Felder (Bild 1 und Tabelle 1).

Speicher-stelle	Bezeichnung	Werte VC 20	C16
43 44	Basic-Beginn	GV : 4096 + 3K : 1024 + 8K : 4608	4096 – –
45 46	Variablen-Beginn	Abhängig von der Programmlänge	
55 56	Basic-Ende	GV : 7680 + 3K : 7680 + 8K : 16384 + 16K : 24576 + 24K : 32768 + 48K : –	16383 – – – – 65535

Tabelle 1. Der Inhalt dieser Speicherzellen bestimmt die Aufteilung des Basic-Bereichs

Der Variablenbereich wächst beim Anlegen neuer Variablen von unten nach oben. Nur das Stringende wandert in entgegenlaufender Richtung.

Die wichtigsten Zeiger, wie unter anderem Beginn und Ende von Basic und Variablen, sind in der Zeropage (Adresse 0 bis 256) abgelegt (Tabelle 1). Dabei ist die Reihenfolge Low-Byte/High-Byte zu beachten (Adresse = High-Byte x 256 + Low-Byte).

Um Speicherplatz für Maschinenprogramme oder Sonderzeichen zu schaffen, hat man prinzipiell zwei Möglichkeiten.

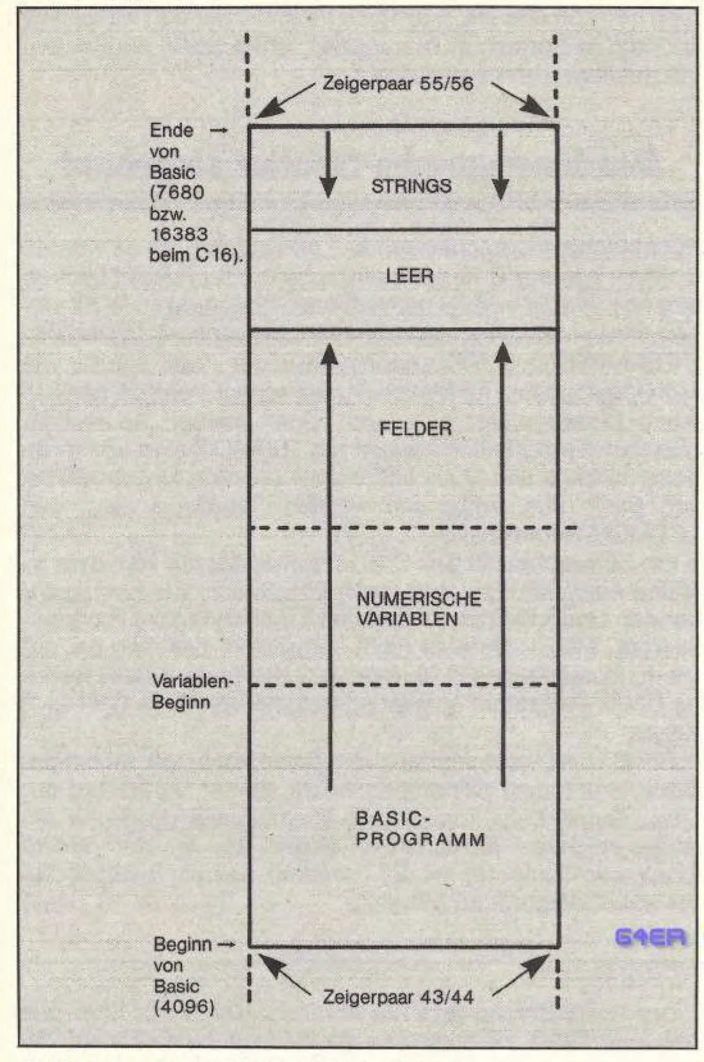


Bild 1. Die Speicherbelegung des Basic-Bereichs

Entweder man verschiebt den Basic-Anfang nach oben oder das Basic-Ende nach unten. Letztere Alternative ist in den meisten Fällen günstiger.

Um zum Beispiel beim VC 20 das Basic-Ende von Adresse 7680 nach 7168 (= 512 Byte) zu verlegen, gibt man ein:

```
POKE 55,0:POKE 56,28:CLR:REM(256 * 28 = 7168)
```

Bei anderen Speichergrößen verfährt man analog.

Der Befehl CLR ist nötig, damit sich verschiedene Hilfszeiger (Stringbeginn und Felderende) anpassen können.

Die andere Alternative der Platzbeschaffung ist etwas komplizierter und gilt sinnvollerweise nur für den VC 20. Sie wird nur bei erweitertem Speicher angewendet, um dort Sonderzeichen abzulegen. Um den Anfang des Programmspeichers von 4608 nach 7680 zu schieben, gibt man:

```
POKE 44,30:POKE 30 * 256,0:NEW
```

ein, denn 30 x 256 ist gerade 7680. Der zweite POKE-Befehl ist nötig, da am Anfang des Basic-Bereichs immer ein Nullbyte stehen muß.

Eine Besonderheit bietet hier der C16 (in der Grundversion). Wenn man mit dem Befehl GRAPHIC die hochauflösende Grafik des C16 einschaltet, dann verkleinert sich der Gesamtspeicher um 14 KByte. In diesem Teil des Speichers (siehe auch Bild 6b) werden dann die grafischen Informationen festgehalten.

Die Reservierung dieses Bereiches wird ebenfalls durch die Veränderung des Zeigerpaares (55/56) realisiert.

Schaltet man in den Textmodus zurück, dann bleiben einem nur noch 2 KByte für Programme. Benötigt man den Platz für die Grafiken nicht mehr, so kann man den dafür freigehaltenen Speicherbereich mit

»POKE 55,0: POKE 56,63: CLR«

fürs Basic wieder freigeben.

Erste Hilfe – Basicprogramme retten nach NEW oder Reset

Schon oft wurden Verfahren beschrieben, um nach einem versehentlichen NEW das Basic-Programm wieder zurückzuholen. Doch wie funktionieren diese Verfahren? Um das zu verstehen, betrachten wir zunächst kurz den Aufbau eines Basic-Programms (Bild 2).

Am Kopf des Programms steht immer eine Null. Dann folgten die Adresse der nächsten Programmzeile (Koppeladresse) und die Zeilennummer. Danach kommt die eigentliche Programmzeile, die sich aus den sogenannten Tokens – den Basic-Codenummern (Tabelle 2) – zusammensetzt. Am Ende dieser Zeile steht dann nochmals eine Null. Die nächste Zeile beginnt wieder mit einem Verbindungszeiger und der Zeilennummer. Das Programm wird mit drei Nullen abgeschlossen. Hieran schließen sich die Variablen an (vergleiche Bild 1).

Durch NEW oder durch einen RESET wird nicht das gesamte Programm, sondern nur der Variablenpointer (45/46) und die erste Koppeladresse gelöscht. Durch Rekonstruktion dieser beiden Zeiger kann das scheinbar verlorene Basic-Programm wieder benutzt werden.

Hier nun das »Rezept« zur Rekonstruktion:

- POKE (Basic-Anfang) + 2,1
  Basic-Anfang in GV = 4096 (4096)
  + 3K = 1024 (-)
  + 8K = 4608 (-)
- SYS 50483:POKE 46,PEEK(35):POKE 45, PEEK (781)+2:CLR
- Die Besitzer des C16 geben statt dessen ein:
  SYS 34840 : SYS 34892

Unbedingt wichtig ist hier die Reihenfolge der Befehle! Ferner darf während der gesamten Prozedur keine Variable definiert werden, da diese das gelöschte Programm überschreiben würde.

Die Funktionsweise ist relativ einfach. Die Unterprogrammroutine in Adresse 50483 bindet die Programmzeilen neu und stellt dabei den ersten Verbindungszeiger wieder her. Sie übergibt dann in den beiden Speicherstellen 35 und 781 die Adresse des Variablenbeginns -2.

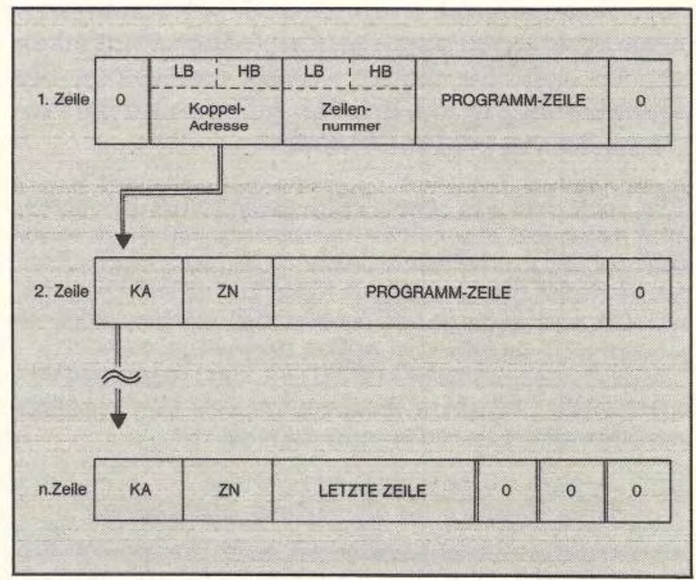


Bild 2. Aufbau eines Basic-Programms. Über die Koppeladressen sind die einzelnen Zeilen miteinander verbunden.

Code (Dezimal)	Zeichen/ Befehl	Code (Dezimal)	Zeichen/ Befehl	Code (Dezimal)	Zeichen/ Befehl
0	Zeilenende	94	↑	189	EXP
1- 31	Leer	95	"	190	COS
32	Space	96-127	Leer	191	SIN
33	!	128	END	192	TAN
34	"	129	FOR	193	ATN
35	#	130	NEXT	194	PEEK
36	$	131	DATA	195	LEN
37	%	132	INPUT#	196	STR$
38	&	133	INPUT	197	VAL
39	'	134	DIM	198	ASC
40	(	135	READ	199	CHR$
41	)	136	LET	200	LEFT$
42	*	137	GOTO	201	RIGHT$
43	+	138	RUN	202	MID$
44	,	139	IF	203	GO
45	—	140	RESTORE	204	RGR
46	.	141	GOSUB	205	RCLR
47	/	142	RETURN	206	RLUM
48	0	143	REM	207	JOK
49	1	144	STOP	208	RDOT
50	2	145	ON	209	DEC
51	3	146	WAIT	210	HEX$
52	4	147	LOAD	211	ERR$
53	5	148	SAVE	212	INSTR
54	6	149	VERIFY	213	ELSE
55	7	150	DEF	214	RESUME
56	8	151	POKE	215	TRAP
57	9	152	PRINT#	216	TRON
58	:	153	PRINT	217	TROFF
59	;	154	CONT	218	SOUND
60	<	155	LIST	219	VOL
61	=	156	CLR	220	AUTO
62	>	157	CMD	221	PUDEF
63	?	158	SYS	222	GRAPHIC
64	@	159	OPEN	223	PAINT
65	A	160	CLOSE	224	CHAR
66	B	161	GET	225	BOX
67	C	162	NEW	226	CIRCLE
68	D	163	TAB(	227	GSHAPE
69	E	164	TO	228	SSHAPE
70	F	165	FN	229	DRAW
71	G	166	SPC(	230	LOCATE
72	H	167	THEN	231	COLOR
73	I	168	NOT	232	SCNCLR
74	J	169	STEP	233	SCALE
75	K	170	+	234	HELP
76	L	171	–	235	DO
77	M	172	*	236	LOOP
78	N	173	/	237	EXIT
79	O	174	↑	238	DIRECTORY
80	P	175	AND	239	DSAVE
81	Q	176	OR	240	DLOAD
82	R	177	>	241	HEADER
83	S	178	=	242	SCRATCH
84	T	179	<	243	COLLECT
85	U	180	SGN	244	COPK
86	V	181	INT	245	RENAME
87	W	182	ABS	246	BACKUP
88	X	183	USR	247	DELETE
89	Y	184	FRE	248	RENUMBER
90	Z	185	POS	249	KEY
91	[	186	SQR	250	MONITOR
92	£	187	RND	251	USING
93	]	188	LOG	252	UNTIL
				253	WHILE

Tabelle 2. Basic-Token des VC 20/C16. Die Codenummern 32 bis 95 entsprechen den normalen ASCII-Zeichen. Nummern größer als 127 sind Token, also Abkürzungen für Basic-Befehle, die der Basic-Interpreter verwendet, um Speicherplatz zu sparen und die Verarbeitungsgeschwindigkeit zu erhöhen. Token ab Code 204 sind nur beim C16 vorhanden.

Blick in die Zeropage

Die Zeropage – oder zu deutsch die Seite null – ist in Maschinensprache besonders einfach zu handhaben. Als Seite bezeichnet man im übrigen immer ein Paket von jeweils 256 Byte. So entspricht Seite 0 den Adressen 0-255, Seite 1 den Adressen 256-511 und so weiter.

Eine Zeropage-Adressierung, zum Beispiel LDA $42, benötigt nur zwei Byte, eine absolute Adressierung, zum Beispiel LDA $1234, hingegen drei Byte im Speicher. Damit verbunden ist auch die Bearbeitungsgeschwindigkeit eines Maschinenprogramms. Denn die Zeropage-Adressierung ist schneller als die entsprechende Drei-Byte-Methode. Bei kleineren Programmen in Assembler fällt dieser Aspekt zwar nicht so sehr ins Gewicht, bei sehr umfangreichen Routinen (wie zum Beispiel beim Basic-Interpreter und bei Schleifen) spielt die Adressierungsart jedoch eine größere Rolle.

In der Seite 0 legt der Computer also insbesondere die Daten ab, die er oft benötigt, wie zum Beispiel Vektoren, Parameter etc. Die komplette Liste der Adreßbelegung zeigt Tabelle 3a (Tabelle 3b gilt für den C16).

Wir wollen es jedoch nicht nur mit der Aufstellung alleine bewenden lassen. Die interessantesten Lokationen möchte ich hier herausgreifen und besprechen. Alle angeführten Adressen gelten, soweit nicht anders angegeben, sowohl für den VC 20 als auch für den C16.

Adresse 43-56: Diese Adressen spielen, wie wir schon im ersten Teil gesehen haben, bei der Verwaltung von Basic-Programmen und -Variablen eine zentrale Rolle. Über sie erfolgt die Trennung zwischen Programm und den einzelnen Variablentypen. Auch für das Speichern und Laden von Programmen liefern sie die Basisdaten. Die einzelnen Funktionen können Sie Tabelle 3 entnehmen.

Eine interessante Gebrauchsmöglichkeit ergibt sich durch die Memoryroutine. Dies ist ein kurzes Basic-Programm, welches die Aufgabe hat, den von Programmen, Variablen, Strings und Arrays belegten Speicherplatz festzustellen (Listing 1):

```
10 rem *** memory dump ***
20 rem
30 a=46:b=44:gosub 100:print"programm:";x
40 a=48:b=46:gosub 100:print"variablen:";x
50 a=50:b=48:gosub 100:print"arrays:";x
60 a=56:b=52:gosub 100:print"strings:";x
70 a=56:b=44:gosub 100:print"speicher:";x
80 print fre(0);"bytes free"
90 end
100 x=peek(a)*256+peek(a-1)-peek(b)*256-pe
    ek(b-1)
110 return
```

Listing 1. »Memory Dump« für VC 20 und C16

Wie man nach dem Starten des Programms sehen kann, erhält man die Werte, indem man die Zeiger – nachdem sie in Dezimalzahlen umgewandelt worden sind – voneinander subtrahiert. Den Speicherplatz, der durch Variablen belegt ist, erhält man beispielsweise durch Subtraktion des Hilfszeigers, »Beginn der Variablen« (45/46) von dem Zeigerpaar »Beginn der Arrays« (47/48).

Gerade bei stark limitiertem Speicherplatz (wie zum Beispiel in der Grundversion) kann dieses Hilfsprogramm Aufschluß über die momentane Speicherverteilung geben.

Adresse 59, 60: Diese Speicherzellen enthalten die laufende Zeilennummer eines Basic-Programms. Sollte es im Programmlauf unterbrochen werden, hat man hier die Möglichkeit, die Zeilennummer nachzulesen.

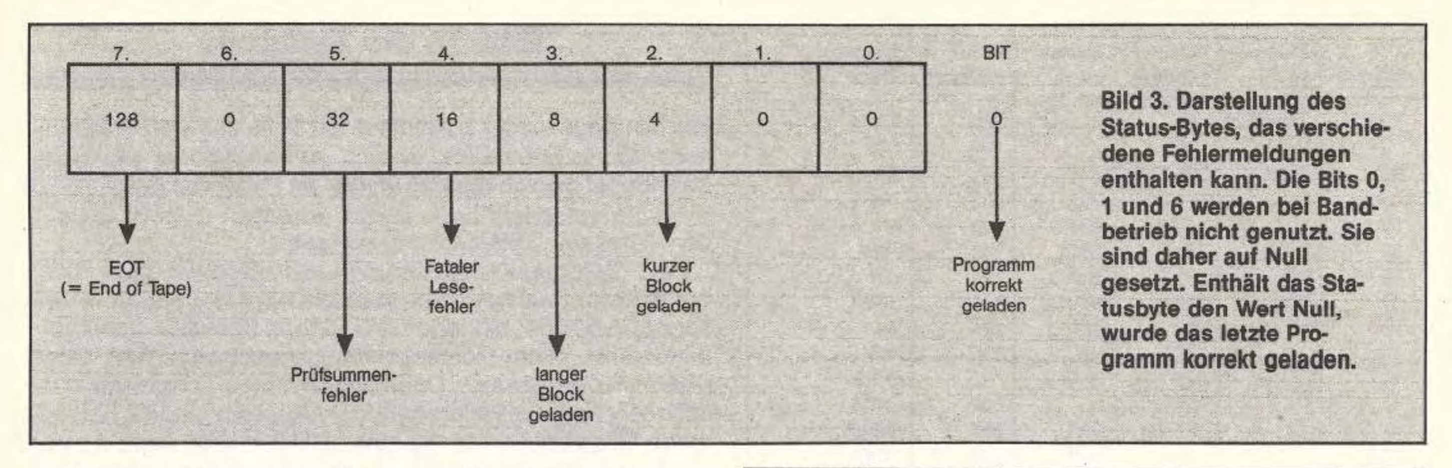


Bild 3. Darstellung des Status-Bytes, das verschiedene Fehlermeldungen enthalten kann. Die Bits 0, 1 und 6 werden bei Bandbetrieb nicht genutzt. Sie sind daher auf Null gesetzt. Enthält das Statusbyte den Wert Null, wurde das letzte Programm korrekt geladen.

Zustandsbeschreibung: das Statusflag

Adresse 144: Dieses Statusflag ist auch von Basic aus über die STATUS- beziehungsweise ST-Anweisung abfragbar. Es liefert das Computerstatus-Byte, dessen Inhalt aufgrund der letzten Input-Output-Operation gesetzt wurde. Bezogen auf den Kassettenport liefert es nach Bild 3 bestimmte Meldungen. Die Informationen über die geladenen Programme werden binär wiedergegeben. Die Null signalisiert ein ordnungsgemäß geladenes Programm. 32 hingegen bedeutet, daß ein Prüfsummenfehler vorliegt. Es ist aber auch möglich, daß der Computer mehrere Meldungen in dieses Byte packt, beispielsweise 52 = 32 + 16 + 4 : Hier wurde ein kurzer Block geladen, jedoch ist die Prüfsumme falsch, und ein fataler Ladefehler liegt vor.

An dieser Stelle ist es angebracht, sich näher mit dem Aufzeichnungsverfahren zu beschäftigen (Bild 4).

So kommen Programme aufs Band

Jeder Speichervorgang beginnt mit dem Header. Dieser Kopf besteht aus dem Vorspann (das ist ein etwa acht Sekunden langer Pfeifton) und dem eigentlichen Programmkopf. Dieser enthält vier wichtige Informationen, nämlich über Programmtyp, Startadresse, Endadresse und Programmname. Diese Daten sind ebenfalls im Bandpuffer zu finden und können von dort abgerufen werden.

Das erste Byte (Adresse 828 beziehungsweise 818 beim C16) gibt Auskunft über den Headertyp. Eine 1 zeigt an, daß es sich um ein Programm handelt, das verschoben geladen werden kann, also auch an eine andere Stelle als die, von der aus es gespeichert wurde. Das Gegenstück dazu ist die absolute Lademethode (Headertyp 3). Gemeint ist »LOAD"..",1,1«. Die Sekundäradresse 1 signalisiert dem Computer, daß er das Programm (unabhängig von den Zeigern 43, 44) wieder in den gleichen Adreßbereich laden soll. Um Verwechslungen vorzubeugen, ist es wichtig, Headertyp und Sekundäradresse zu unterscheiden. Für den Headertyp gibt es drei Möglichkeiten:

1. Laden mit Verschiebelader (die Anfangsadresse wird durch den Zeiger 43 und 44 bestimmt).
2. Ein File - also Daten aus Variablen - wurde gespeichert.
Die Unterscheidung ist wichtig, denn Daten können nicht mit LOAD geladen werden.
3. Ein Programm ist absolut zu laden.

Die nächsten vier Byte geben die Anfangs- und Endadresse des geladenen Files an. Mit der Endadresse hat es eine besondere Bewandtnis. Sie wird nämlich nach korrektem Laden der Zeropage (Adresse 45, 46) übergeben. Bei

Dezimal	Hexadezimal	Bemerkung
0 - 2	00 - 02	USR-Sprungvektor (Normal: JMP $D248)
3 - 4	03 - 04	Vektor für Unterprogramm 'Fließkomma nach Integer'
5 - 6	05 - 06	Vektor für Unterprogramm 'Integer nach Fließkomma'
7	07	Suchzeichen (sucht ':' oder Zeilenende)
8	08	Hochkommaflag
9	09	Spaltenspeicher beim TAB-Befehl
10	0A	LOAD/VERIFY Flag (0 = LOAD/1 = VERIFY)
11	0B	Eingabepufferzeiger (Anzahl der Elemente)
12	0C	Flag für Dim (enthält die laufende Variable)
13	0D	Variablentyp (0 = Numerisch/128 = String)
14	0E	Numerische Variable (0 = Fließkomma/128 Integer)
15	0F	Flag bei DATA und LIST
16	10	Flag für FN
17	11	Eingabeflag (0 = INPUT/64 = GET/152 = READ)
18	12	Vorzeichen bei ATN
19	13	Aktuelles Ein-/Ausgabegerät
20 - 21	14 - 15	Integerwert (zum Beispiel Zeilennummer
22	16	Zeiger im Stringstapel
23 - 24	17 - 18	Zeiger auf den zuletzt verwendeten String
25 - 33	19 - 21	Stringstapel
34 - 37	22 - 25	Speicher für div. Hilfszeiger
38 - 42	26 - 2A	Speicherbereich bei einer Multiplikation
43 - 44	2B - 2C	Zeiger auf Beginn des Basic-Speicherbereichs
45 - 46	2D - 2E	Beginn der Variablen (= Ende des Programms)
47 - 48	2F - 30	Beginn der Arrays (= Ende der Variablen)
49 - 50	31 - 32	Zeiger auf Ende der Arrays
51 - 52	33 - 34	Zeiger auf Beginn der Strings
53 - 54	35 - 36	Hilfszeiger für Strings
55 - 56	37 - 38	Zeiger auf Basic-Ende
57 - 58	39 - 3A	Momentane Basic-Zeilennummer
59 - 60	3B - 3C	Vorherige Basic-Zeilennummer
61 - 62	3D - 3E	Zeiger auf lfd. Basic-Befehl (für CONT)
63 - 64	3F - 40	Momentane Zeilennummer für DATA
65 - 66	41 - 42	Momentane Adresse einer DATA-Zeile
67 - 68	43 - 44	Zeiger auf Element in DATA-Zeile und INPUT-Vektor
69 - 70	45 - 46	Momentaner Variablenname
71 - 72	47 - 48	Adresse der momentanen Variablen
73 - 74	49 - 4A	Zeiger für FOR/NEXT Variable
75 - 76	4B - 4C	Zwischenspeicher für Programmzeiger (bei SAVE)
77	4D	Speicher für Vergleichssymbole
78 - 79	4E - 4F	Zeiger für FN
80 - 83	50 - 53	Verschieden genutzter Speicherbereich
84 - 86	54 - 56	FN-Sprungvektor (ähnlich USR)
87 - 96	57 - 60	Feld für diverse arithmetische Zwecke (Arithmetik Akku #3 und #4)
97	61	Fließkommaakku #1 : Exponent
98 - 101	62 - 65	Fließkommaakku #1 : Matisse
102	66	Fließkommaakku #1 : Vorzeichen
103	67	Zeiger für Polynomauswertung
104	68	Überlauf von Akku #1
105	69	Fließkommaakku #2 : Exponent
106 - 109	6A - 6D	Fließkommaakku #2 : Mantisse
110	6E	Fließkommaakku #2 : Vorzeichen
111	6F	Vergleichsbyte der Vorzeichen von Akku 1 und 2
112	70	Rundung für Akku #1 und #2

Tabelle 3a. Die Zeropage beim VC 20

Dezimal	Hexadezimal	Bemerkung
113 - 114	71 - 72	Länge des Kassettenpuffers
115 - 138	73 - 8A	CHRGET-Routine
139 - 143	8B - 8F	RND-Wert als Fließkommazahl
144	90	Statusflag ST
145	91	STOP-Flag
150	96	Kassetten EOT (End of Tape) erhalten
151	97	Zwischenspeicher für Register
152	98	Anzahl der offenen Dateien
153	99	Eingabegerät (Normal = 0 : Tastatur)
154	9A	Ausgabegerät (Normal = 3 : Bildschirm)
155	9B	Paritätsbyte)
150	96	Kassetten EOT (End of Tape) erhalten
151	97	Zwischenspeicher für Register
152	98	Anzahl der offenen Dateien
153	99	Eingabegerät (Normal = 0 : Tastatur)
154	9A	Ausgabegerät (Normal = 3 : Bildschirm)
155	9B	Paritätsbyte vom Band (Prüfsumme)
156	9C	Flag für Byte erhalten (von Band)
157	9D	Flag für Direktmodus (= 128) oder Programm (= 0)
158	9E	Band: erster Teil - Prüfsumme
159	9F	Band: zweiter Durchlauf - Prüfsumme
160 - 162	A0 - A2	Interne Uhr (Stunde, Minute, Sekunde)
163	A3	Bitzähler für serielle Ausgabe
164	A4	Zähler für Band
165	A5	Startsynchronisation bei Kassetten
166	A6	Zeiger im Bandpuffer
167 - 171	A7 - AB	Flags für Schreiben/Lesen bei Band
172 - 173	AC - AD	Zeiger auf Kassettenpuffer und für scrolling
174 - 175	Ae - AF	Zeiger auf Programmende bei LOAD
176 - 177	B0 - B1	Bandzeitkonstanten
178 - 179	B2 - B3	Startadresse des Bandpuffers
180	B4	Bitzähler für Band
181	B5	Band oder RS232: nächstes zu sendendes Bit
182	B6	Übertragungsspeicher für RS232
183	B7	Länge des Filenamens
184	B8	Logische Filenummer
185	B9	Sekundäradresse
186	Ba	Gerätenummer
187 - 188	BB - BC	Zeiger auf Filenamen
189	BD	Ein-/Ausgabespeicher (für seriell)
190	BE	Blockzähler für Band
191	BF	Puffer für serielle Ausgabe
192	C0	Bandmotor Flag
193 - 194	C1 - C2	Startadresse für Ein-/Ausgabe
195 - 196	C3 - C4	Endadresse für Ein-/Ausgabe
197	C5	gedrückte Taste (momentaner Wert im Tastaturmatrixcode)
198	C6	Anzahl der gedrückten Tasten (im Tastaturpuffer)
199	C7	Bildschirm: negative Anzeige
200	C8	Zeiger auf Zeilenende bei Eingabe
201	C9	Cursorzeile
202	CA	Cursorspalte
203	CB	Welche Taste? (64 = keine)
204	CC	Cursorblinken
205	CD	Cursor Blinkzähler
206	CE	Zeichen unter dem Cursor
207	CF	Flag für Cursorblinken (0 = An/1 = Aus)
208	D0	Eingabe von Bildschirm/oder Tastatur
209 - 210	D1 - D2	Start der aktuellen Bildschirmzeile
211	D3	Position des Cursors in der aktuellen Bildschirmzeile
212	D4	Flag für Cursor (0 = Direkt, sonst programmiert)
213	D5	Länge der Bildschirmzeile (21 od. 42 od. 63 od. 84)
214	D6	Cursorzeile
215	D7	Zeiger für diverse Zwecke
216	D8	Anzahl der Inserts
217 - 242	D9 - F2	High Bytes der Bildschirmzeilenanfänge
243 - 244	F3 - F4	Zeiger im Farbspeicher
245 - 246	F5 - F6	Zeiger in der Tastaturdekodiertabelle
247 - 248	F7 - F8	Zeiger auf RS232 Eingabepuffer
249 - 250	F9 - FA	Zeiger auf RS232 Ausgabepuffer
251 - 254	FA - FE	Freie Zeropageadressen
255 - (266)	FF - 10A	Zwischenspeicher für Fließkomma nach ASCII

Tabelle 3a. Die Zeropage beim VC 20 (Schluß)

Dezimal	Hexadezimal	Bemerkung
0	$0000	Datenrichtungsregister des 7501
1	$0001	Ein-/Ausgabe-Port des 7501
2	$0002	Flag für Schleifen
3- 4	$0003-0004	Neue Startadresse (Renumber)
5- 6	$0005-0006	Schrittweite (Renumber)
7	$0007	Gesuchtes Zeichen
8	$0008	Flag für Anführungszeichen-Modus
9	$0009	TAB-Spaltenzähler
10	$000A	Flag: 0 = Load, 1 = Verify
11	$000B	Zeiger für Eingabepuffer, Anzahl der Elemente
12	$000C	Flag für Standard-DIM
13	$000D	Datentyp: $FF=String, $00=Numerisch
14	$000E	Datentyp: $80=Integer, $00=Fließkomma
15	$000F	Flag für DATA/LIST
16	$0010	Flag: Element/FNx-Flag
17	$0011	Flag: $00=INPUT, $40=GET, $98=READ
18	$0012	Flag: Vorzeichen des ATN
19	$0013	Flag: Input-Prompt
20- 21	$0014-0015	Int. Adresse
22	$0016	Zeiger auf temporären Stringstapel
23- 24	$0017-0018	Letzter temporärer Stringvektor
25- 33	$0019-0021	Stapel für temporäre Strings
34- 35	$0022-0023	Bereich für Hilfszeiger 1
36- 37	$0024-0025	Bereich für Hilfszeiger 2
38	$0026	Bereich für Produkt bei Multiplikation
39	$0027	Bereich für Produkt bei Multiplikation
40	$0028	Bereich für Produkt bei Multiplikation
41	$0029	Bereich für Produkt bei Multiplikation
42	$002A	Bereich für Produkt bei Multiplikation
43- 44	$002B-002C	Zeiger auf Basic-Anfang
45- 46	$002D-002E	Zeiger auf Variablen-Anfang
47- 48	$002F-0030	Zeiger auf Beginn der Arrays
49- 50	$0031-0032	Zeiger auf Ende der Arrays (+1)
51- 52	$0033-0034	Zeiger auf Stringspeicher
53- 54	$0035-0036	Hilfszeiger für Strings
55- 56	$0037-0038	Zeiger auf Speichergrenze
57- 58	$0039-003A	Laufende Basiczeilennummer
59- 60	$003B-003C	Textpointer
61- 62	$003D-003E	Zeiger auf Basic-Statement für CONT
63- 64	$003F-0040	Nummer der aktuellen DATA-Zeile
65- 66	$0041-0042	Adresse des aktuellen DATA-Elementes
67- 68	$0043-0044	Sprungvektor für INPUT
69- 70	$0045-0046	Aktueller Variablenname
71- 72	$0047-0048	Adresse der aktuellen Variablen
73- 74	$0049-004A	Variablenzeiger für FOR...NEXT
75- 76	$004B-004C	Zwischenspeicher für Basic-Zeiger
77	$004D	Akkumulator für Vergleichssymbole
78- 79	$004E-004F	Arbeitsbereich (Zeiger etc.)
80- 81	$0050-0051	Arbeitsbereich (Zeiger etc.)
82	$0052	Arbeitsbereich (Zeiger etc.)
83	$0053	Grafik-Modus
84- 86	$0054-0056	Sprungvektor für Funktionen
87- 96	$0057-0060	Bereich für numerische Operationen
97	$0061	Fließkomma-Akkumulator 1 (FAC): Exponent
98-101	$0062-0065	Fließkomma-Akkumulator 1 (FAC): Mantisse
102	$0066	Fließkomma-Akkumulator 1 (FAC): Vorzeichen
103	$0067	Zeiger für Polynom-Auswertung
104	$0068	Fließkomma-Akkumulator 1 Überlauf
105-110	$0069-006E	Fließkomma-Akkumulator 2 Exponent etc.
111	$006F	Vorzeichenvergleich Akku 1 mit Akku 2
112	$0070	Fließkomma-Akkumulator 1 niederwertige Stelle
113-114	$0071-0072	Kassettenpuffer Länge/Zeiger
115-116	$0073-0074	Automatisches Zeileninkrement 0=AUS
117	$0075	Grafik-Flag 0=Text 255=Grafik
118-120	$0076-0078	Arbeitsbereich
121-123	$0079-007B	Darstellung von DS$
124-125	$007C-007D	Basic-Pseudo-Stack-Pointer
126-143	$007E-008F	Arbeitsbereich (Sound)
144	$0090	Statuswort ST

Tabelle 3b. Die Zeropage beim C16

Dezimal	Hexadezimal	Bemerkung
145	$0091	Flag: Stoptaste / RVS-Taste
148	$0094	Flag: Serieller Bus — Ausgabe Zeichenpuffer
149	$0095	Zeichenspeicher — Serieller Bus
150	$0096	Zwischenspeicher
151	$0097	Anzahl der geöffneten Files
152	$0098	Eingabegerät (Normalwert: 0)
153	$0099	Ausgabegerät (Normalwert: 3)
154	$009A	Flag: $80=Direkt-, $00=Programm-Modus
157-158	$009D-009E	Zeiger auf Ende des Programms
159-160	$009F-00A0	Temporärer Datenspeicher
161-162	$00A1-00A2	Temporärer Datenspeicher
163-165	$00A3-00A5	Echtzeit-Uhr
166	$00A6	Register für seriellen Bus
167	$00A7	Register für Kassettenroutine
168	$00A8	Register für seriellen Bus
169	$00A9	Temporärer Farb-Vektor
170	$00AA	Register für Kassettenroutine
171	$00AB	Anzahl der Zeichen im Filenamen
172	$00AC	Aktuelle logische File-Nummer
173	$00AD	Aktuelle Sekundär-Adresse
174	$00AE	Aktuelle Geräte-Nummer
175-176	$00AF-00B0	Zeiger auf Filenamen
177	$00B1	Von Fehlerroutine benutzt
178-179	$00B2-00B3	I/O Startadresse
180-181	$00B4-00B5	Basis-Ladeadresse
182-183	$00B6-00B7	Zeiger: Anfang Kassettenpuffer
186-187	$00BA-00BB	Zeiger auf Zeichen im Kassettenpuffer
190-191	$00BE-00BF	Register für Long-Fetch-Routine
192-193	$00C0-00C1	Register für Scrolling
194	$00C2	RVS-Flag (Bildschirm)
195	$00C3	Zeiger: Ende der Zeile (Input)
196-197	$00C4-00C5	Cursorposition (Input), Reihe/Spalte
198	$00C6	Tastaturabfrage ($40=keine Taste)
199	$00C7	Flag: Eingabe Bildschirm/Tastatur
200-201	$00C8-00C9	Zeiger auf Bildschirmzeile
202	$00CA	Cursorposition in aktueller Bildschirmzeile
203	$00CB	Flag: Anführungszeichen-Modus ($00=nein)
204	$00CC	Länge der aktuellen Bildschirmzeile
205	$00CD	Zeile, in der sich der Cursor befindet
206	$00CE	Letztes Zeichen (I/O)
207	$00CF	Anzahl der Zeichen (Insert-Modus)
208-215	$00D0-00D7	Reserviert für Software
216-232	$00D8-00E8	Reserviert für Anwendungssoftware
233	$00E9	Arbeitsbereich
234-235	$00EA-00EB	Bildschirm-Editor (Farbe)
236-238	$00EC-00EE	Arbeitsbereich (Bildschirm)
239	$00EF	Anzahl der Zeichen im Tastaturpuffer
241-242	$00F1-00F2	Register für Monitor
250	$00FA	Register für X bei Stoptastentest
251	$00FB	Aktuelle Bank-Konfiguration
254	$00FE	Arbeitsregister (Editor)

Tabelle 3b. Die Zeropage beim C16 (Schluß)

Load Error geschieht dies nicht; »PRINT FRE(0)« zeigt dann die volle Bytezahl an, obwohl sich ein Programm im Speicher befindet. Man darf das Programm – das vielleicht nur einen geringfügigen Fehler hat – dann nicht starten, weil die Variablen dieses überschreiben würden. Abhilfe schafft in diesem Fall

```
POKE 45,PEEK(831):POKE 46,PEEK(832):CLR.
```

Für den C16 gibt man ein:

```
POKE 45,PEEK(821):POKE 46,PEEK(822): CLR
```

Die Werte werden damit »von Hand« übertragen und ein normaler Programmablauf ist in den meisten Fällen wieder möglich.

Jetzt aber wieder zurück zu Bild 3: Nachdem der Vorspann und der Programmkopf vor einem Trennzeichen (dies ist ein ganz kurzer Pieps) auf Band geschrieben worden sind, wird der Header gleich noch einmal gespeichert.

Dies ist eine Eigenheit des Commodore-Systems, das der Datensicherheit dient. Denn nachdem der Programmkopf 1 geladen hat, vergleicht er in einem zweiten Durchgang das bisher Geladene (welches sich ja schon im Speicher befindet) mit Programmkopf 2. Eine Abweichung veranlaßt den Computer, eine Fehlermeldung auszugeben, in ganz schweren Fällen wird der Ladevorgang gleich ganz unterbrochen. Diese Verfahrensweise gilt nicht nur für den Programmkopf, sondern auch für Programme und Daten – sie alle werden doppelt gespeichert.

Beim C16 kommt noch eine spezielle Eigenheit hinzu. Da der recht komplexe TED-Chip (siehe separaten Artikel in diesem Heft) den Prozessor bei dessen Ladearbeit – die ja sehr schnell vonstatten gehen muß – stark behindert, wird er ganz einfach abgeschaltet. Aus diesem Grund bleibt der Bildschirm während des Ladevorganges dunkel.

Nachdem also der Programmkopf erkannt worden ist, zeigt der Computer den Filenamen an, und das eigentliche Programm (oder die Daten) wird geladen. Ihnen ist wiederum ein kurzer Vorspann vorangestellt. Der Endblock besteht aus Endmarkierung und einem Nachspann. Er zeigt dem Computer das Ende des geladenen Programms an, womit der Ladevorgang beendet ist.

Nach diesem Exkurs zum Kassettenaufzeichnungsformat nun wieder zur Zeropage.

Adresse 145: Mit Hilfe dieser Speicherstelle kann man den Zustand der STOP- und der linken SHIFT-Taste abfragen: Wert =

253: Linke SHIFT-Taste gedrückt
254: STOP-Taste gedrückt
255: Keine der beiden gedrückt

Wenn man das Low-Byte des STOP-Vektors (Adresse 808,

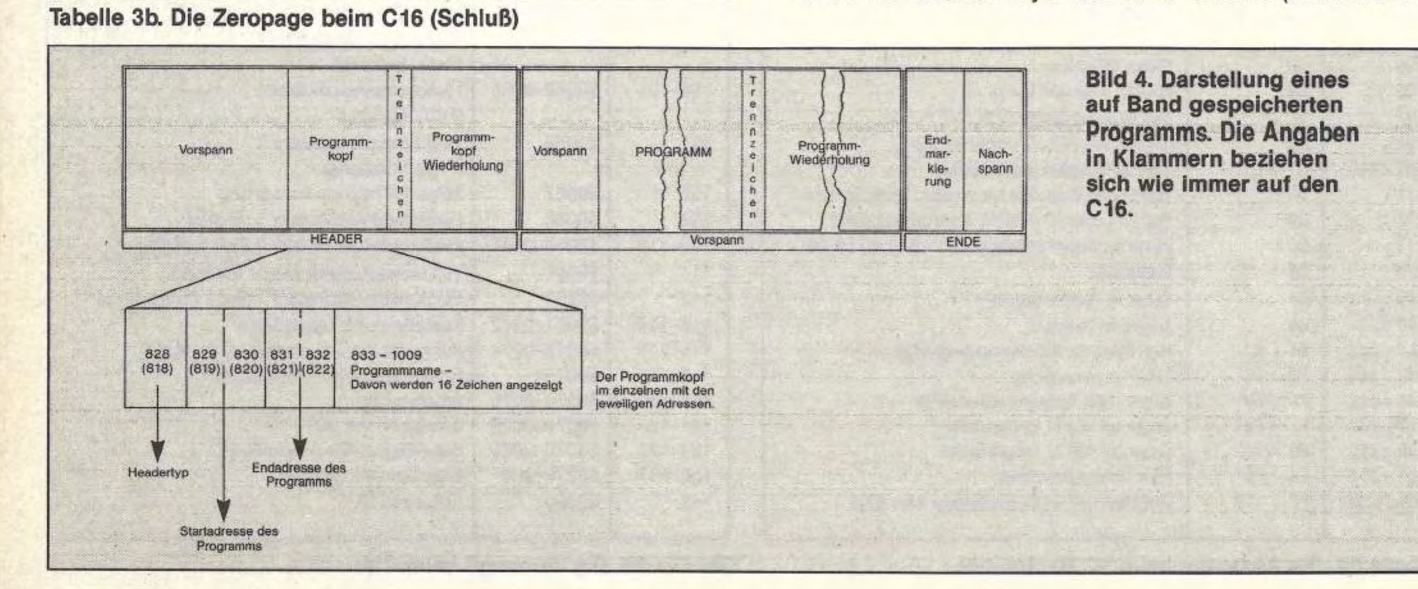


Bild 4. Darstellung eines auf Band gespeicherten Programms. Die Angaben in Klammern beziehen sich wie immer auf den C16.

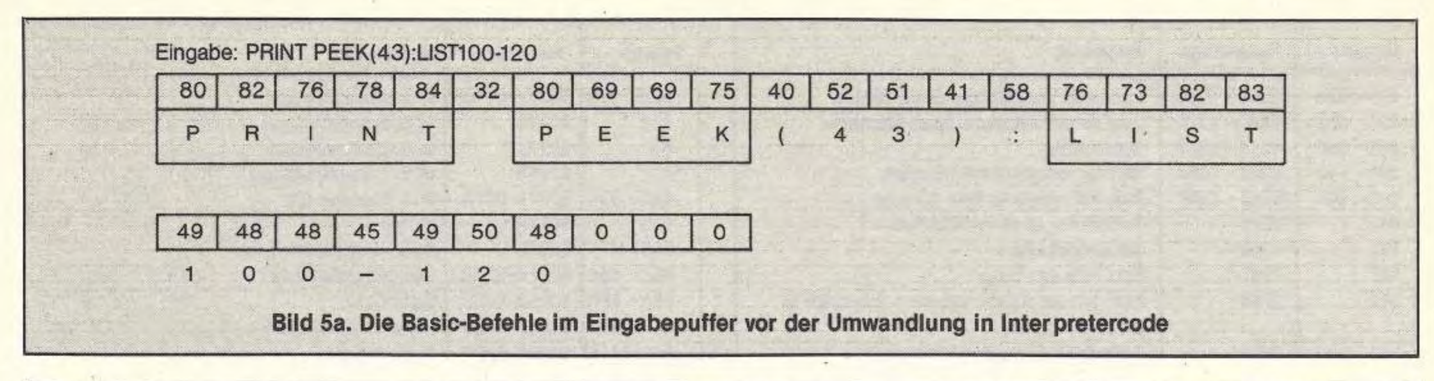


Bild 5a. Die Basic-Befehle im Eingabepuffer vor der Umwandlung in Interpretercode

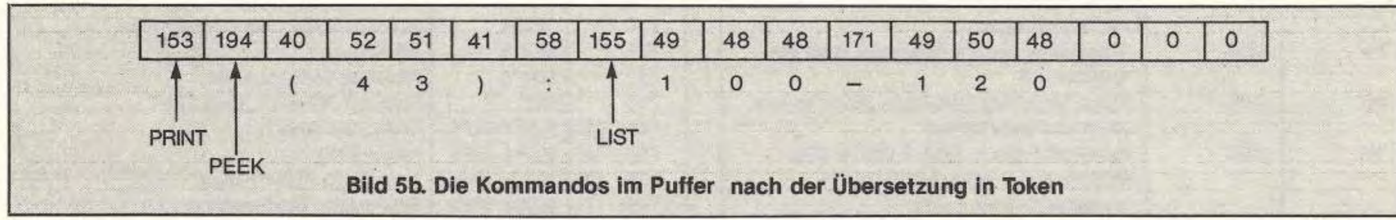


Bild 5b. Die Kommandos im Puffer nach der Übersetzung in Token

beim C16: 806) mit POKE 808,PEEK (808)-2 ändert (bitte beim C16 Adresse 806 statt 808 verwenden), so ist man in der Lage, diese Taste von Basic aus abzufragen, ohne das laufende Programm anzuhalten. Ihr kann dann in der Routine eine besondere Funktion zugewiesen werden, beispielsweise Anhalten des Programmablaufs für eine bestimmte Zeit.

Der Weg einer Eingabezeile

Wenn wir einen Basic-Befehl im Direktmodus (LIST, RUN, PRINT) oder eine Programmzeile mit Zeilennummer eingeben, werden die Informationen – wie bekannt – auf den Bildschirm geschrieben und gelangen gleichzeitig in den Basic-Eingabepuffer (Adresse 512–600/ $0200–$0258). Dort werden Programmzeilen oder direkte Befehle zunächst einmal im ASCII-Format gesammelt (Bild 5a). Dies geschieht solange, bis man die RETURN-Taste betätigt.

Dieser Puffer hat eine Kapazität von 88 Zeichen – also den bekannten vier Bildschirmzeilen beim C20 beziehungsweise 2 Bildschirmzeilen beim C16 .

Drückt man die RETURN-Taste, so beginnt der Interpreter mit der Auswertung der Kommandos. Eingaben ohne Zeilennummer werden auf ihre Syntax hin überprüft und danach ausgeführt.

Verfolgen wir nun einmal genauer den Weg einer Befehlszeile. Die einzelnen ASCII-Zeichen werden von einer Maschinensprachen-Unterroutine namens »CHRGET« aus dem Puffer gelesen und mit den Befehlswörtern aus dem ROM verglichen. War die Überprüfung positiv – ist der Befehl also identifiziert worden –, so verkürzt der Computer die Kommandozeile, indem er die Befehle in Token (Tabelle 2) umwandelt. Diese Prozedur durchlaufen sowohl die Programmzeilen (mit Zeilennummer) als auch die direkten Kommandos (Bild 5b). An dieser Stelle trennen sich nun aber die Wege dieser beiden Zeilentypen.

Zunächst zu dem weiteren Weg einer Programmzeile. Die inzwischen komplett übersetzte Zeile im Basic-Puffer wird nun in einem zweiten Durchlauf vom Zwischenspeicher in den Programmspeicher übertragen, wobei sie auch gleich richtig eingeordnet wird (damit die Reihenfolge der Zeilennummern stimmt). Weil sich dadurch der Programmbereich im Basic-Speicher vergrößert, muß der Variablenbereich weiter oben angesiedelt werden. Die bis dahin gespeicherten Variablen werden dadurch natürlich überschrieben. Nach dem Übertragen der Programmzeile springt das Interpreterprogramm wieder in die Warteschleife zurück, damit weitere Befehle entgegengenommen werden können.

Nun möchte ich den weiteren Weg einer Direktmoduszeile beschreiben, denn der verläuft anders. Nachdem die Zeile mit Hilfe der CHRGET-Routine (Adresse 115–138 beziehungsweise beim C16: 1139–1156) in Interpretercode (also Token) umgewandelt worden ist, wird der 2-Byte-Zeiger ($7A–$7B) auf den Pufferanfang zurückgestellt und der Inhalt wie eine Programmzeile behandelt und abgearbeitet (wieder mit Hilfe der CHRGET-Routine).

Verbotenes

Die Befehle INPUT und GET dürfen im Direktmodus nicht verwendet werden. Der Grund liegt darin, daß diese Eingabebefehle ebenfalls den Basic-Eingabepuffer zur Zwischenspeicherung der Daten verwenden. Das »Direktmodusprogramm«, das sich zu dieser Zeit im Puffer befindet, würde dann überschrieben. Um das zu verhindern, sind diese Kommandos im Direktmodus verboten (Fehlermeldung »ILLEGAL DIRECT«).

Der nächste größere Komplex, den wir hier behandeln wollen, ist die Tastaturverwaltung. Wie Sie bestimmt schon öfter bemerkt oder gelesen haben, wickelt der VC 20 (und der C16) seine Tastaturoperationen ebenfalls über einen Puffer ab (vergleiche Tabelle 4). Dies wird beim LISTen von Basic-Programmen deutlich, denn während der Computer ein Programm ausgibt, können alle Tasten (mit Ausnahme des STOP-Keys) drücken – diese erscheinen aber nicht auf dem Bildschirm. Erst wenn das Programm zu Ende gelistet ist, sieht man, daß keine Taste »vergessen« wurde, denn alle Eingaben wurden im Tastaturpuffer (Adresse 631–640, beim C16: 1319–1328) zwischengespeichert.

Mit Hilfe des Tastaturpuffers kann aber auch – und das ist das Interessante an diesen zehn Bytes – eine relativ unkonventionelle Art der Programmierung praktiziert werden.

Dezimal	Hexadezimal	Bemerkung
255- 266	00FF - 010A	Arbeitsspeicher für Fließkomma nach ASCII
256- 318	0100- 013E	Korrekturpuffer für Bandbetrieb
256- 511	0100- 01FF	Prozessor Stack
512- 600	0200- 0258	Basic-Eingabepuffer
601- 610	0259- 0262	Tabelle der logischen Filenummern...

Tabelle 4a. Die Belegung der Seiten 2 bis 4 beim VC20

Dezimal	Hexadezimal	Bemerkung
611- 620	0263 - 026C	....sowie der dazugehörigen Gerätenummern...
621- 630	026D - 0276	..und der entsprechenden Sekundäradressen
631- 640	0277 - 0280	Tastaturpuffer
641- 642	0281 - 0282	Start des verfügbaren RAM-Bereichs
643- 644	0283 - 0284	Ende des verfügbaren RAM-Bereichs
645	0285	Timeout-Flag für den seriellen Port
646	0286	Aktueller Farbcode
647	0287	Farbe unter dem Cursor
648	0288	High-Byte des Bildschirmspeichers (Information für das Betriebssystem)
649	0289	Größe des Tastaturpuffers (Maximum 10)
650	028A	Repeat-Flag (0: Cursor + Space/ 64: Keine Taste/ 128: Alle Tasten mit Repeat)
651	028B	Repeat-Zähler (bestimmt die Wartezeit bis die Taste wiederholt wird)
652	028C	Repeat-Verzögerung (bestimmt die Zeit, bis die Taste das erstemal wiederholt wird)
653	028D	Kontrolltasten-Flag (1: SHIFT/ 2: CBM/ 4: CTRL. Es können auch 2 Tasten erkannt werden zum Beispiel 5: SHIFT + CTRL)
654	028E	Letzte Kontrolltaste (Identisch mit 653)
655- 656	028F - 0290	Vektor für Tastaturdecodierung
657	0291	Flag für SHIFT + CBM gesperrt (keine Groß-Klein-Schrift-Umschaltung. 0: Normal/128: Sperre)
658	0292	Flag für Scrolling
659- 670	0293 - 029E	RS 232 Register, Zeiger, etc.
671- 672	029F - 02A0	Zwischenspeicher für IRQ bei Bandbetrieb
673- 676	02A1 - 02A4	Diverse VIA-Zeiger
677- 767	02A5 - 02FF	Zwischenspeicher einer Bildschirmzeile
786- 769	0300 - 0301	Vektor für Fehlermeldung (C43A)
770- 771	0302 - 0303	Vektor für Basic-Warmstart (C483)
772- 773	0304 - 0305	Vektor für Umwandlung von ASCII in Token (C579)
774- 775	0306 - 0307	Vektor für Umwandlung von Token in ASCII (C717)
776- 777	0308 - 0309	Vektor für Basic-Befehlsadresse (C7E1)
778- 779	030A - 030B	Vektor für arithmetisches Element (CE83)
780	030C	Akku für SYS-Befehl (Bei SYS wird 780 in den Akku geladen)
781	030D	X-Reg für SYS-Befehl
782	030E	Y-Reg für SYS-Befehl
783	030F	Speicher für Status Register für SYS-Befehl
788- 789	0314 - 0315	IRQ-Vektor (EABF) *** Kernal-Vektoren
790- 791	0316 - 0317	BRK-Vektor (FED2)
792- 793	0318 - 0319	NMI-Vektor (FEAD)
794- 795	031A - 031B	OPEN-Vektor (F40A)
796- 797	031C - 031D	CLOSE-Vektor (F34A)
798- 799	031E - 031F	Kanal für Eingabe (CHKIN, F2C7)
800- 801	0320 - 0321	Kanal für Ausgabe (CHOUT, F309)
802- 803	0322 - 0323	Kanäle initialisieren (CLRCH, F3F3)
804- 805	0324 - 0325	Eingabe-Vektor (F20E)
806- 807	0326 - 0327	Ausgabe-Vektor (F27A)
808- 809	0328 - 0329	Vektor für STOP-Taste prüfen (F770)
810- 811	032A - 032B	GET-Vektor (F1F5)
812- 813	032C - 032D	Alle Files schließen (F3EF)
814- 815	032E - 032F	User-Vektor (FED2)
816- 817	0330 - 0331	LOAD—Vektor (F549)
818- 819	0332 - 0333	SAVE-Vektor (F685)
828-1019	033C - 03FB	Kassettenpuffer

Tabelle 4a. Die Seiten 2 bis 4 beim VC 20

Dezimal	Hexadezimal	Bemerkung
256- 511	$0100-01FF	Prozessorstack
512- 600	$0200-0258	Eingabepuffer
601- 604	$0259-025C	Basic-Puffer
605- 684	$025D-02AC	Basic-/DOS-Arbeitsbereich
605	$025D	DOS-Schleifenzähler
606- 621	$025E-026D	Bereich für Filenamen
622	$026E	1. Filename (Länge)
623	$026F	DOS (Laufwerk 1)
624- 625	$0270-0271	1. Filename (Adresse)
626	$0272	2. Filename (Länge)
627	$0273	DOS (Laufwerk 2)
628- 629	$0274-0275	2. Filename (Adresse)
630	$0276	DOS logische Adresse
631	$0277	DOS (Geräteadresse)
632	$0278	DOS (Sekundäradresse)
633- 634	$0279-027A	DOS (Disketten-ID)
635	$027B	ID-Flag
636	$027C	DOS (Ausgabepuffer)
637- 684	$027D-02AC	DOS (Arbeitsbereich)
685- 688	$02AD-02B0	Grafik-Cursor
689- 692	$02B1-02B4	Grafik-Cursor (Register)
693- 715	$02B5-02CB	Grafik-Register
716- 739	$02CC-02E3	Print-Using, Grafik-Arbeitsbereich
740	$02E4	High-Byte-Adresse des Charakter-ROM
747	$02EB	Trace-Flag
752	$02F0	Anzahl der Grafik-Parameter
753	$02F1	Parameter: Relativ (1), Absolut (0)
754- 755	$02F2-02F3	Fließkomma-Vektor
756- 757	$02F4-02F5	Integer-Vektor
768- 769	$0300-0301	Sprungvektor: Fehlermeldung
770- 771	$0302-0303	Sprungvektor: Basic-Warmstart
772- 773	$0304-0305	Sprungvektor: Token-Generierung
774- 775	$0306-0307	Sprungvektor: Keywort erzeugen
776- 777	$0308-0309	Sprungvektor: Hauptschleife
778- 779	$030A-030B	Sprungvektor: Eval
780- 781	$030C-030D	Sprungvektor: Token-Generierung (User)
782- 783	$030E-030F	Sprungvektor: Keywort erzeugen
784- 785	$0310-0311	Sprungvektor: User-Token bearbeiten
786- 787	$0312-0313	Sprungvektor: Interrupt (Uhr)
788- 789	$0314-0315	Sprungvektor: Interrupt
790- 791	$0316-0317	Sprungvektor: Break Interrupt
792- 793	$0318-0319	Sprungvektor: Open
794- 795	$031A-031B	Sprungvektor: Close
796- 797	$031C-031D	Sprungvektor: Kanal für Eingabe öffnen
798- 799	$031E-031F	Sprungvektor: Kanal für Ausgabe öffnen
800- 801	$0320-0321	Sprungvektor: I/O zurücksetzen
802- 803	$0322-0323	Sprungvektor: Input
804- 805	$0324-0325	Sprungvektor: Ausgabe
806- 807	$0326-0327	Sprungvektor: Abfrage der Stoptaste
808- 809	$0328-0329	Sprungvektor: Getin-Routine
810- 811	$032A-032B	Sprungvektor: Schließen aller Files
812- 813	$032C-032D	Sprungvektor: Monitor Break
814- 815	$032E-032F	Sprungvektor: Lade-Routine
816- 817	$0330-0331	Sprungvektor: Save-Routine
818-1010	$0332-03F2	Kassettenpuffer
1011-1012	$03F3-03F4	Datenzähler (Write)
1013-1014	$03F5-03F6	Datenzähler (Read)
1015-1078	$03F7-0436	RS-232-Input-Puffer (64 Byte)
1079-1108	$0437-0454	System-Arbeitsbereich
1109-1138	$0455-0472	System-Arbeitsbereich
1139-1144	$0473-0478	Chrget-Routine
1145-1156	$0455-0484	Chrgot-Routine
1172-1244	$0494-04DC	Bankswitching-Routinen
1255-1258	$04E7-04EA	Print-Using-Parameter
1263-1270	$04EF-04F6	Trap- und Error-Flags
1280-1282	$0500-0502	USR-Sprungbefehl
1283-1287	$0503-0507	Startwert für RND
1288	$0508	Flag für Kalt- oder Warmstart
1289-1298	$0509-0512	Tabelle der logischen Filenummern
1299-1308	$0513-051C	Tabelle der Gerätenummern
1309-1318	$051D-0526	Tabelle der Sekundäradressen
1319-1328	$0527-0530	Tastaturpuffer
1329-1330	$0531-0532	Basic-Anfang
1331-1332	$0533-0534	Basic-Ende
1337	$0539	Zeiger: Kassettenpuffer
1338	$053A	Typ des Kassettenfiles
1344	$0540	Tasten-Widerholfunktion $80=alle, $40=keine, $00=nur DEL, CUR, SPC
1345-1346	$0541-0542	Zähler für Wiederholfunktion
1347	$0543	Shift-Flag
1348	$0544	Letztes Shift-Zeichen
1349-1350	$0545-0546	Sprungvektor: Keylog
1351	$0547	Text-/Grafik-Flag

Tabelle 4b. Die Belegung der Seiten 2 bis 4 beim C16

Dezimal	Hexadezimal	Bemerkung
1352	$0548	Scroll-Flag
1355-1372	$054B-0551	Cpu-Arbeitsbereich
1362-1367	$0552-0557	Cpu-Register
1368-1372	$0558-055C	Cpu-Arbeitsbereich
1373	$055D	Zähler für Funktionstasten
1374	$055E	Zeiger: Text, Funktionstasten
1375-1510	$055F-05E6	Speicher für Funktionstasten
1516-1571	$05EC-06EB	1 Page, Bankingroutinen
1516-1519	$05EC-05EF	Adreßtabelle
1520-1521	$05F0-05F1	Long-Jump (Adresse)
1522	$05F2	Long-Jump (Akkumulator)
1523	$05F3	Long-Jump (X-Register)
1524	$05F4	Long-Jump (Status-Register)
1525-1629	$05F5-065D	RAM-Bereich für Bankswitching
1630-1771	$065E-06EB	RAM-Bereich für Benutzersoftware
1792-1967	$06EC-07AF	Basic-Pseudo-Stack
1968-1996	$07B0-07CC	Kassetten-Arbeitsbereich
1997-2000	$07CD-07D0	RS-232-Arbeitsbereich
2001	$07D1	RS-232: Pointer auf Anf. Eingabepuffer
2002	$07D2	RS-232: Pointer auf Ende Eingabepuffer
2003	$07D3	Anzahl der Zeichen im Eingabepuffer
2021	$07E5	Bildschirm: Unterer Rand
2022	$07E6	Bildschirm: Oberer Rand
2023	$07E7	Bildschirm: Linker Rand
2024	$07E8	Bildschirm: Rechter Rand
2026	$07EA	$FF=Automatisches Einfügen
2030-2033	$07EE-07F1	Bit-Tabelle
2034	$07F2	A-Register retten bei Sys-Kommando
2035	$07F3	X-Register retten bei Sys-Kommando
2036	$07F4	Y-Register retten bei Sys-Kommando
2037	$07F5	Status-Register retten bei Sys-Kommando
2038	$07F6	Register für Tastaturabfrage
2039	$07F7	Flag: CTRL-S: $00=offen, $06=gesperrt
2040	$07F8	RAM-ROM-Umschaltung für Monitor 0=ROM, $80=RAM
2044	$07FC	Motorsteuerung

Tabelle 4b. Die Seiten 2 bis 4 beim C16 (Schluß)

```
1 REM DATA-ERZEUGER
2 REM
5 B=248:REM 'B=208' BEI C 16
10 INPUT"{CLR,2DOWN}STARTADRESSE:";SA
20 INPUT"{DOWN}ENDADRESSE:";EA
30 A=SA:GOSUB 250:POKE B,A1:POKE B+1,A2
40 A=EA:GOSUB 250:POKE B+2,A1:POKE B+3,A2
50 POKE B+4,44:POKE B+5,1:REM STARTZEILE 3
   00
60 DEF FN Z(X)=PEEK(X)+256*PEEK(X+1)
65 B=248:REM 'B=208' BEI C 16
70 AD=FN Z(B+4):REM AKTUELLE ZEILENNUMMER
80 PRINT"{CLR}";AD;"DATA ";
90 REM ** DATA-ZEILE MIT 8 ELEMENTEN ERZEU
   GEN **
100 FOR T=0 TO 7
105 X=PEEK(FN Z(B)+T)
110 B$=STR$(X):L=LEN(B$)
120 B$=RIGHT$(B$,L-1)
130 IF L=4 THEN 150
140 FOR Y=L+1 TO 4:B$="0"+B$:NEXT
150 B$=","+B$
160 IF T=0 THEN B$=RIGHT$(B$,3)
170 PRINT B$;:IF FN Z(B)+T=FN Z(B+2) THEN
    240
180 NEXT
190 A=AD+10:GOSUB 250:POKE B+4,A1:POKE B+5
    ,A2
200 A=FN Z(248)+8:GOSUB 250:POKE B,A1:POKE
     B+1,A2
210 PRINT:PRINT"{DOWN}RUN 60"
215 REM TASTATURPUFFER MIT <HOME>,<RETURN>
    ,<CRSR DOWN> UND <RETURN> FUELLEN
220 Z=631:REM 'Z=1319' BEI C 16
225 POKE Z,19:POKE Z+1,13:POKE Z+2,17:POKE
     Z+3,13
230 POKE 198,4:REM 'POKE 239,4' BEI C 16
240 END
250 A2=INT(A/256):A1=A-A2*256
260 RETURN
```

Listing 2. »Data-Erzeuger« für VC 20 (für C16 REM-Zeilen beachten)

Um dies verständlich zu machen, habe ich ein Programm (Listing 2) geschrieben, welches DATA-Zeilen aus Maschinenprogrammen oder Sonderzeichen erzeugt. Die Problematik dabei ist folgende: Wenn ich DATA-Zeilen ins Programm schreiben möchte, so ist dies nur im Direktmodus möglich. Um dies automatisch zu tun, benötigen wir ein Programm. Also was tun? – Man bedient sich des Tastaturpuffers!

Der »DATA-Erzeuger«

Dazu nochmals die Durchleuchtung der Funktionsweise. Während des Systeminterrupts, der alle 60stel Sekunde durchlaufen wird, fragt der Computer die Tastatur ab. Ein eingegebenes Zeichen wird dabei im Tastaturpuffer abgelegt, wo es so lange verbleibt, bis ein Zeichen von der Tastatur benötigt wird. Dies ist zum Beispiel im Direktmodus der Fall (wenn der Cursor blinkt) oder im Programm – bei INPUT oder GET. Die Zeichen werden dann wieder aus dem Tastaturpuffer hervorgeholt, und zwar nach dem Prinzip »First in, First out«. Bei unserem Programm werden zunächst einmal sechs Speicherzellen initialisiert. Die ersten zwei Adressen enthalten die Anfangsadresse der abzuspeichernden Daten aus dem Speicher. Dieser Zeiger wird solange inkrementiert, bis er den Wert der Endadresse – der in den zwei folgenden Bytes abgelegt ist – erreicht hat. In den letzten zwei Speicherstellen (Adresse 252, 253 beziehungsweise 212, 213) steht die Zeilennummer in der Reihenfolge Low-/High-Byte. Bequemer wäre es natürlich, wenn man normale Variablen verwenden könnte. Dies ist jedoch nicht möglich, weil diese beim Einfügen einer DATA-Zeile gelöscht werden. Darum bleibt nur der Umweg über Speicherstellen, deren Inhalte durch Basic nicht überschrieben werden können.

Als nächstes erzeugt das Programm – mit Hilfe der in 252/253 beziehungsweise 212/213 gespeicherten Zwei-Byte-Zahl – eine Zeilennummer, welche auf den Bildschirm gePRINTet wird. Dann schreibt es das Befehlswort »DATA« und druckt acht dreistellige Zahlen (die Daten aus dem zu verarbeitenden Maschinenprogramm) aus. Nun haben wir eine fertige Programmzeile auf dem Bildschirm stehen, die sich allerdings noch nicht im Speicher befindet. Dies erledigt jetzt unser Tastaturpuffer.

Vorher müssen wir uns jedoch genau überlegen, wie unser Bildschirm aussieht, denn dementsprechend muß der Cursor programmiert werden.

Zuerst wird der Cursor mit HOME (=CHR$ (19)) an die linke obere Ecke befördert. Dort wird durch einen Druck auf die RETURN-Taste (=CHR$(13)) die Basic-Zeile in den Speicher übernommen. Nun befindet sich der Cursor in der dritten Zeile. Durch ein »Cursor down« (=CHR$(17)) bewegt er sich eine Zeile nach unten und steht nun auf dem Befehlswort RUN 60. Ein weiteres RETURN bewirkt den erneuten Start des Hilfsprogramms.

Wir benötigen für die Cursorbewegung also vier Werte. Diese werden vor dem Ende des Programms mit »`POKE 631,19: POKE 632, 13: POKE 633,17: POKE 634, 13: POKE 198,4`« (beziehungsweise die oben angegebenen Werte für den C16) in den Tastaturpuffer geschrieben. Hierdurch simulieren wir eine gedrückte Tastenfolge: denn nachdem sich der

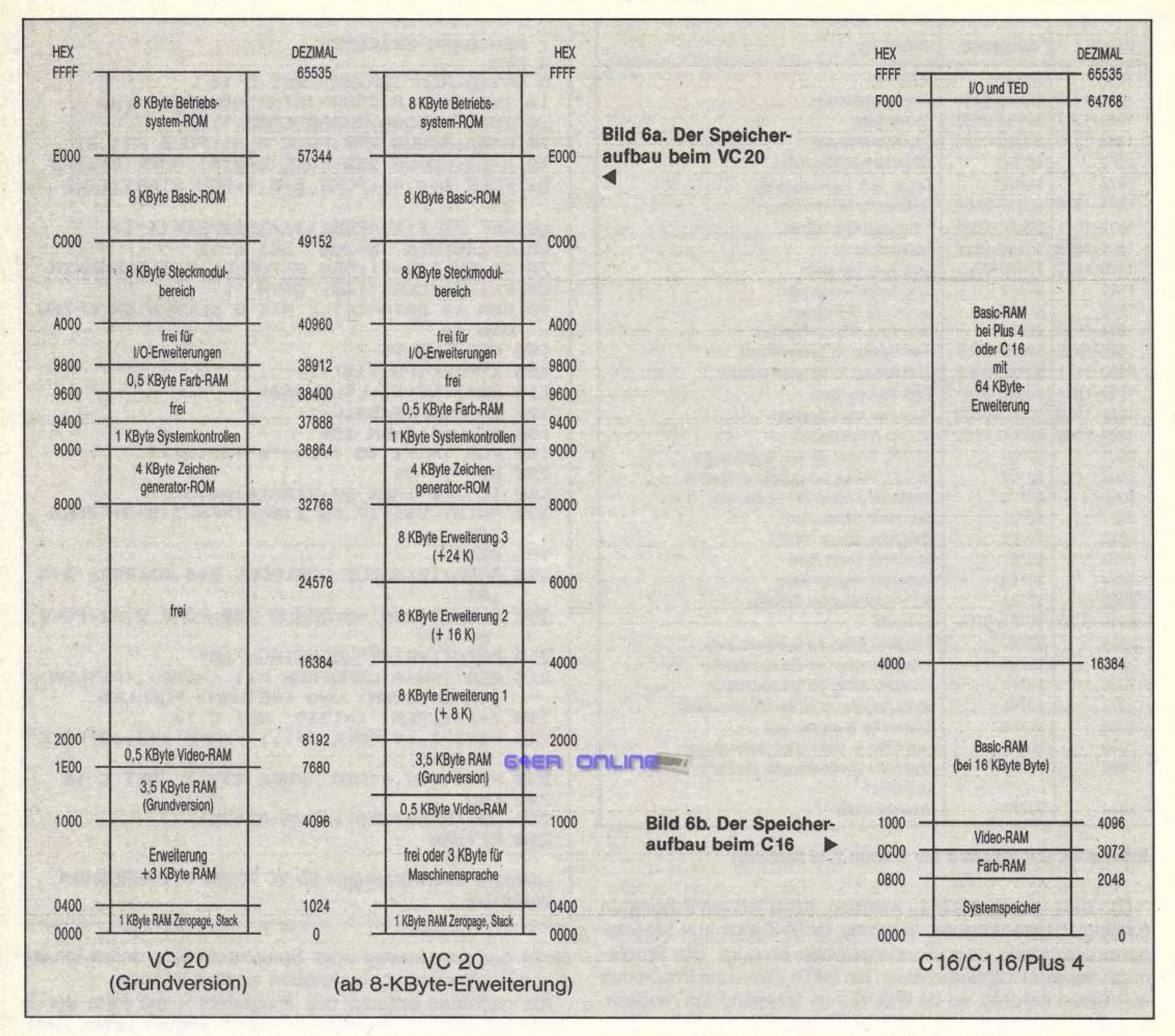


Bild 6a. Der Speicheraufbau beim VC 20 ◀

Bild 6b. Der Speicheraufbau beim C16 ▶

Computer über den Befehl END wieder im Direktmodus befindet, wird zuerst der Tastaturpuffer geleert, wodurch der Cursor den vorbestimmten Weg nimmt. »POKE 198,4« (»POKE 239,4«) gibt an, wie viele Zeichen sich momentan im Puffer befinden.

Jetzt möchte ich noch einen kleinen Einblick in die Speicherorganisation geben, was auch im Hinblick auf Grafik von Bedeutung ist.

Bei Erweiterungen von mehr als 8 KByte verändert sich die Lage des Bildschirm- und Farbspeichers (die Einstellung nimmt die Reset-Routine vor). Anhand von Bild 6a soll erläutert werden, warum eine Verschiebung notwendig ist.

Der **V**ideo-**I**nterface-**C**hip VIC (daher auch der englische Name des VC 20), der vor allem für die Erzeugung des Fernsehsignals und den Aufbau des Bildschirmes verantwortlich ist, kann hardwaremäßig nur Videospeicherplätze zwischen 4096 und 8192 adressieren. Folglich muß das Bildschirm-RAM in diesem Bereich angesiedelt werden.

In der Grundversion liegt es zwischen Adresse 7680 und 8191 – also am Ende des verfügbaren Speichers, damit der Speicherbereich für Basic auch bei eingesteckter 3 KByte-Erweiterung durchgängig ist (läge der Bildschirmspeicher so wie bei einer 8-KByte-Erweiterung, wäre dies nicht der Fall).

Ist ein Speichermodul von mehr als 8 KByte eingesteckt (egal ob der 3-KByte-Bereich zugeschaltet ist oder nicht), so legt das System den Videospeicher an die unterste adressierbare Stelle für den VIC, also Adresse 4096. Aus diesem Grund kann die eingesteckte 3-KByte-Erweiterung nicht mehr für Basic benutzt werden, denn sonst wäre der Speicher nicht mehr durchgängig.

Der C16 ist vom Speicheraufbau etwas anders organisiert. Die in beiden Geräten eingebauten Prozessoren (6502 und 7501) können jeweils immer nur einen Adreßbereich von 64 KByte (das entspricht den Adressen 0–65535) ansprechen. Um Zugriff auf noch mehr Speicherplatz für Programme und Daten (sei es nun im ROM oder im RAM) zu haben, bedient man sich beim C16 eines Tricks:

Der Computer schaltet einfach immer zwischen zwei Speicherblöcken zu je 64 KByte (siehe Bild 6b) hin und her. Der erste Block besteht – wenn der C16 voll ausgebaut ist – aus Schreib-/Lesespeicher, die andere Bank verwaltet das ROM. Diese Technik des »Bank-Switching« wird auch beim neuen Commodore-Flaggschiff C 128 verwendet.

Damit ist unsere Einführung in die internen Geheimnisse des VC 20/C16 beendet. Es ist jetzt an Ihnen, die hier erworbenen Kenntnisse durch eigenes Experimentieren zu vertiefen und in eigenen Programmen einzubauen. Zusätzliche wichtige Hintergrund-Informationen speziell über die Themengebiete »Grafik« und »Zeichensatz« beim C16 finden Sie in diesem Sonderheft. (Christoph Sauer/ev)



»60671 BYTES FREE« beim C16

Wenn Sie größere Programme erstellen und auch noch im Grafik-Modus arbeiten, dann sind Sie sehr schnell am Ende des Speicherbereichs angelangt. Wir haben für Sie zwei Speichererweiterungen getestet, die da Abhilfe schaffen.

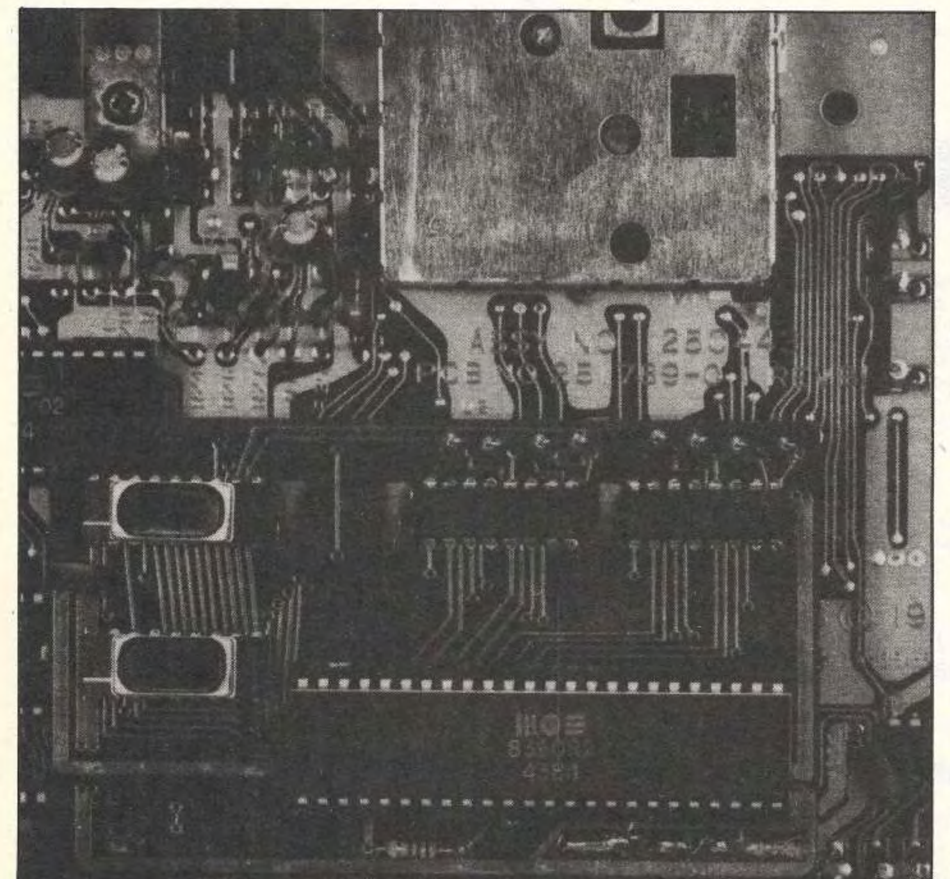
Haben Sie ihren C16 eingeschaltet, so meldet er sich unter anderem mit »12277 BYTES FREE«. Für die ersten Programmierversuche reicht ein Speicherplatz von 12 KByte auch vollkommen aus. Wenn Sie allerdings schon einige Erfahrungen gesammelt haben und sich in die Grafik-Programmierung vorwagen, wird dies schlagartig anders. Bei eingeschaltetem Grafik-Modus stehen Ihnen nämlich nur noch 2 KByte für das Programm zur Verfügung. Das bedeutet, nach etwa 40 vollgeschriebenen Programmzeilen ist Schluß. Sollten Sie auf die Idee kommen, am Anfang des Programms noch einige Variablenfelder zu dimensionieren, brauchen Sie gar nicht weiterzuprogrammieren.

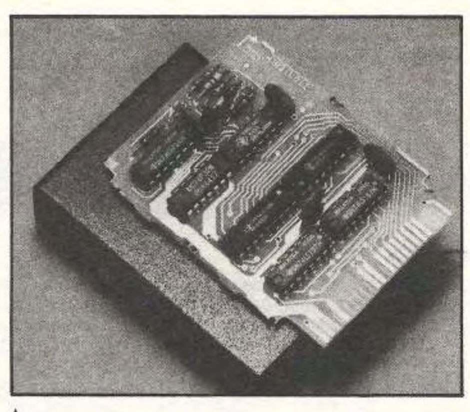
Die Firma Kingsoft setzt dieser Misere mit zwei Speichererweiterungen ein Ende. Sie haben die Wahl, Ihren Commodore 16 auf 32 KByte oder sogar 64 KByte aufzustocken. Wollen wir uns zunächst der kleineren von beiden Versionen zuwenden.

Die 16 KByte Erweiterung (Bild 1) wird bei ausgeschaltetem Gerät einfach in den Expansion-Port eingesteckt. Die geschlossene Gehäuseseite muß dabei oben sein. Das Modul läßt sich zwar nicht verkehrt einstecken, aber um Ungewißheiten beim Käufer zu beseitigen, sollte dies auch in der sonst guten Einbauanleitung erwähnt werden.

16 KByte Erweiterung

»28661 BYTES FREE« ist nach dem Einschalten auf dem Bildschirm zu sehen. Damit haben Sie auch im Grafik-Modus einen größeren Speicherbereich zum Programmieren frei als vorher ohne Grafik-Aktivierung. Auch der fortgeschrittene Programmierer verfügt jetzt wieder über genügend Speicherplatz. Etwa 360 Programmzeilen können Sie jetzt von vorne bis hinten vollschreiben.





▲
Bild 1. 16 KByte Erweiterung. Bei ausgeschaltetem Gerät in den Expansion-Port stecken. Die geschlossene Gehäuseseite muß dabei nach oben zeigen.

◄
Bild 2. Erweiterung auf 64 KByte. Diese Erweiterung muß in den Computer eingebaut werden. Beachten Sie dazu die im Text beschriebenen Hinweise.

Gut sind die mitgelieferten Tips, denn es kann vorkommen, daß einige Programme mit der Erweiterung nicht richtig laufen. Einige POKE-Befehle schaffen da Abhilfe:
statt: LOAD jetzt: POKE 65299,17 : LOAD
statt: RUN jetzt: POKE 65299,17 : RUN
Nicht nur bei dieser Erweiterung kann der geschilderte Trick helfen. Es gilt ebenso für die 64 KByte-Erweiterung und wenn C16-Programme auf dem Plus 4 laufen sollen.

Ein Mangel zeigte sich bei dem Modulgehäuse. Da die Platine nur durch kleine Nasen im Gehäuse gehalten wird, kann es bei mechanischer Belastung des Gehäuses vorkommen, daß die Platine aus den besagten Halterungen springt.

Mit 99 Mark kann man den Preis für diese Erweiterunng als durchaus akzeptabel bezeichnen.

Erweiterung auf 64 KByte

Eine Einschaltmeldung von »60671 BYTES FREE« läßt das Herz des Programmierers höher schlagen. Der C64 kann da mit »38911 BYTES FREE« fast als kleiner Bruder bezeichnet werden. Über Platzmangel für Ihr Programm können Sie sich jetzt auf alle Fälle nicht mehr beklagen.

Das dafür notwendige Erweiterungsmodul muß allerdings in den Computer eingebaut werden (Bild 2). Eine Arbeit, die Ihnen aber mit Hilfe der beigelegten Einbauanleitung leicht fallen wird. Sie müssen dafür den Computer aufschrauben (3 Schrauben auf der Rückseite des Gerätes), die Stecker von der Tastatur und der LED für Netzspannungskontrolle abziehen (merken Sie sich, wie sie eingesteckt waren) und die Abschirmung zurückklappen. Jetzt liegt das Innenleben des Computers offen vor Ihnen. Das größte IC auf der ganzen Platine ist der Video Interface Controller mit der Typenbezeichnung 8360. Mit einem Schraubendreher, den Sie zwischen Sockel und IC ansetzen, können Sie diesen Baustein vorsichtig nach oben abhebeln. Um ein Verbiegen der IC-Beinchen zu vermeiden, sollten Sie dies abwechselnd von der linken und rechten Seite des ICs tun. Den Video Interface Controller stecken Sie jetzt in den IC-Sockel der Zusatzplatine. Achten Sie unbedingt darauf, daß dabei die Kerbe am IC und die ebenfalls eingekerbte Sockelseite übereinanderliegen müssen. Auf die gleiche Weise stecken Sie die Zusatzplatine in den freien Sockel des Computers. In der Anleitung werden Sie auch darauf aufmerksam gemacht, daß beim Aufstecken der Zusatzplatine einige Bauteile im Weg sein könnten und diese vorsichtig umgebogen werden müssen. Wir wollen Sie aber trotzdem darauf hinweisen, daß ein folgenschwerer Kurzschluß entstehen kann, falls Sie diese Anweisung nicht befolgen und das Gerät einschalten.

Bedenken Sie jedoch, daß die Garantieansprüche durch diesen Eingriff erlöschen. Dennoch ist diese Speichererweiterung ein sehr interessantes Angebot. Für 199 Mark können Sie den Speicher Ihres C16 auf 64 KByte aufrüsten und damit alle Möglichkeiten des Computers intensiv nutzen.

Kingsoft bietet noch mehr

Kingsoft hat einiges für C16-Besitzer auf Lager und gehört damit zu den wenigen Firmen, die sich bisher um diesen Computer gekümmert haben. Neben je einem Zeichen- und Musikprogramm werden zirka 50 Spiele angeboten. Eine recht interessante Kleinigkeit ist ein Joystickadapter für 12 Mark, mit dem Sie C64-Joysticks an Ihren Computer anschließen können. (kn)

Info: Kingsoft, Fritz Schäfer, Schnackebusch 4, 5106 Roetgen

Drucker für C16

Lohnt sich der Kauf eines Druckers für den C 16? Wir stellen Ihnen sechs Drucker unter 800 Mark zur Auswahl vor.

Der Kauf eines Druckers sollte wohl überlegt sein, denn mit einem falschen Drucker sind schnell mehrere 100 Mark aus dem Fenster geworfen. Dabei ist es keineswegs gleichgültig, für welchen Drucker einer Preisgruppe man sich entscheidet, denn Leistungen und Druckerbild sind von Gerät zu Gerät so unterschiedlich, daß man sich vor dem Kauf über die Vor- und Nachteile genauestens informieren sollte. Wir haben für Sie einige Low-Cost-Drucker in der Preislage unter 800 Mark getestet.

Kleiner Epson ganz groß

Was zunächst wegen seiner kompakten Maße wie ein verkleinertes Modell aussieht, erweist sich in der Praxis schnell als ein recht vielseitiges und zuverlässiges Werkzeug. Der Epson P-40 (Bild 1) ist ein Kleindrucker mit einer Papierbreite von 11,2 Zentimetern, der ursprünglich für den HX-20 Hand-Held-Computer entwickelt wurde. Er arbeitet nach dem Thermo-Prinzip und ist deswegen extrem leise. Im flachen Gehäuse präsentiert er sich wie eine verkleinerte Ausgabe des bekannten FX-80-Druckers. In der Tat teilt der P-40 mit seinem großen Bruder nicht nur die Centronics-Schnittstelle,

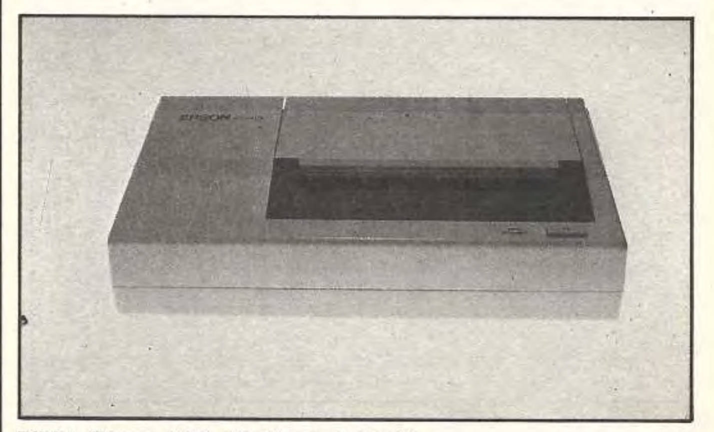


Bild 1. Epson P-40, ein Thermo-Drucker

```
EPSON P40
PAPIERBREITE: 11 ZENTIMETER
ZEICHENMATRIX: 5 X 9
DRUCKGESCHWINDIGKEIT: 40
ZEICHEN PRO SEKUNDE
GRAFIKFAEHIG: JA, ZWEI PUNKTDICHTEN
DOPPELTE BREITE
VERSCHIEDENE ZEILENABSTAENDE
KOMPRIMIERTE SCHRIFT
HERVORGEHOBENE SCHRIFT
DEUTSCH UMLAUTE:
AÖÜ^äöüß
```
(Schriftbild verkleinert)

Bild 2. Klein, aber mit Leistungen der Großen ausgestattet — der Epson P-40

sondern auch einige Steuerbefehle. Für einen Drucker dieser Preisklasse ungewöhnlich sind Befehle zum Einstellen der verschiedenen Schriftarten (Bild 2) wie komprimierter (80 Zeichen) und gedehnter (40 Zeichen) Schrift. Wer das gut strukturierte und umfangreiche Handbuch studiert, stößt sogar auf zwei Befehle für einfache und doppelte Grafik, die den Befehlen der »großen« Brüder entsprechen. Ganz erstaunlich ist auch der eingebaute Zeichengenerator. Er bietet die Möglichkeit zwischen verschiedenen internationalen Zeichensätzen zu wählen, unter anderem auch einem deutschen. Dieser Zeichensatz enthält 96 ASCII-Zeichen, inklusive der Groß- und Kleinschreibung. Trotz seines guten Konzeptes ist der Betrieb des P-40 am C 16 nicht ohne Probleme. In jedem Fall wird ein zusätzliches Interface notwendig, das mindestens 50 Mark kostet. Für eine riesige Auswahl solcher Schnittstellen ist allerdings gesorgt: Da der P-40 über die gleiche Befehlssyntax wie seine »großen Brüder« verfügt, können alle für die RX-80/FX-80 konstruierten Schnittstellen verwendet werden. Mittels Batterien kann er auch ohne Netzteil betrieben werden. Insgesamt ist der P-40 ein gelungenes Gerät, bei dem es allerdings am rechten Einsatzgebiet fehlt. Für eine Textverarbeitung ist sein Papier zu schmal und als Listingdrucker fehlt ihm der Commodore-Zeichensatz. Mit einem Preis von 448 Mark (ohne Interface) ist der P-40 auch etwas teuer.

Doppeltes Lottchen

Der Brother HR-5 (Bild 3) ist ein Thermo-Transfer-Drucker, den es in zwei verschiedenen Ausführungen gibt. An einem »C« hinter dem Namen erkenntlich, stellt sich die direkt an den C 16 anschließbare Version vor. Ein eingebautes Interface sorgt für alle Anpassungen, die für den Betrieb am C 16 wichtig sind. Beim HR-5 ohne »C« stehen zwei Schnittstellen, Centronics parallel oder V.24 (RS232C) zur Verfügung. Das Druckverfahren des HR-5C ist etwas ungewöhnlich. Während des Druckes fährt der Druckkopf am stillstehenden Farbband entlang und preßt es gegen das Papier. Dabei werden die angesteuerten Punkte auf dem Druckkopf erwärmt und die Farbpartikel bleiben auf dem Papier hängen. Nach dem Druck einer Zeile hält der Druckkopf an und es wird mit verhältnismäßig lautem Geräusch das Farbband weitergespult. Da der Druck bidirektional abläuft, wird die nächste Zeile in der Regel von rechts nach links gedruckt. Beim HR-5C können zwei verschiedene Papiersorten verwendet werden. Neben dem Druck auf normalem Papier mit Farbband (Bild 4) kann der Drucker auch direkt auf Thermopapier drucken. Man hat also die Wahl zwischen teurem Farbband und billigem Papier oder teurem Thermopapier.

Der HR-5C ist zum Betrieb mit vier Monozellen vorgesehen. Wahlweise kann auch ein Netzgerät, das aber mit 40 Mark extra bezahlt werden muß, verwendet werden. Diese Anschaffung ist aber ratsam, denn bei einer Leistungsaufnahme von 6 Watt sind Batterien natürlich schnell erschöpft.

Sehr viel Fingerfertigkeit verlangt das Einstellen der unter der Führungsstange und dem Steuerriemen verborgenen DIL-Schalter und das Einlegen der Farbbandkassette. Das eingebaute Interface wurde in wesentlichen Punkten an die Steuerung der Commodore-Drucker angepaßt. Mit der Sekundäradresse 0 erfolgt der Ausdruck im Normalmodus (Großbuchstaben und Grafikzeichen). Mit der Sekundäradresse 7 erreicht man den Zeichensatz mit großen und kleinen Buchstaben. Zu den Fähigkeiten des HR-5C gehört auch der Druck von reversen und vergrößerten Zeichen. Das umfangreiche Handbuch erleichtert die Einarbeitung in den HR-5C.

Der HR-5C ist mit einem Preis von 298 Mark sicherlich kein schlechter Kauf für alle, denen es auf problemlosen Anschluß und niedrigen Geräuschpegel ankommt. Drei Dinge sind es aber, die den sonst guten Eindruck des HR-5C schmälern: Die relativ hohen Unterhaltskosten, die niedrige Druckgeschwindigkeit (10 bis 30 Zeichen pro Sekunde) und die unpraktische Handhabung.

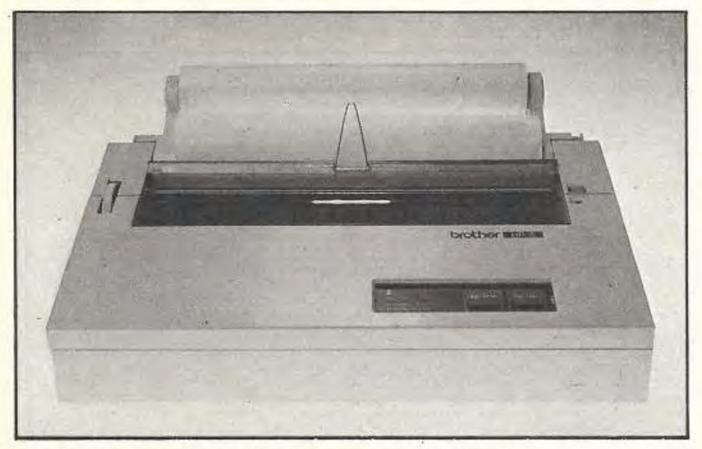


Bild 3. Brother HR-5, ein Thermo-Transfer-Drucker

```
BROTHER HR 5 THERMOTRANSFERDRUCKER
PAPIERBREITE: 21 ZENTIMETER
ZEICHENMATRIX: 9X9
DRUCKGESCHWINDIGKEIT 30
ZEICHEN PRO SEKUNDE
GRAFIKFAEHIG: JA, ZWEI PUNKTDICHTEN
DOPPELTE BREITE
VERSCHIEDENE ZEILENABSTAENDE
KOMPRIMIERTE SCHRIFT
HERVORGEHOBENE SCHRIFT
UNTERSTRICHENE SCHRIFT
ELITE SCHRIFT MIT DEM HR-5
DEUTSCH UMLAUTE:
ÄÖÜ^äöüß
```

(Schriftbild verkleinert)

Bild 4. Gute Leistung und fast nicht zu hören – der Thermo-Transfer-Drucker Brother HR-5

Die Hausmarke

Seikosha und Commodore-Drucker sind zwei Worte für den gleichen Begriff. Drucker dieser Firma haben den C 64, und vorher den VC 20, auf weiten Strecken ihrer Entwicklung begleitet. Seikosha war auch die erste Firma, die außer Commodore selbst, direkt an den seriellen Port des C 16 anschließbare Drucker anbot. Kein Wunder, denn die Commodore 1525- und MPS 801-Drucker werden eigentlich von Seikosha produziert (nur das Gehäuse stammt von Commodore). So kommt es auch, daß einige unserer Testkandidaten auffällige Ähnlichkeiten besitzen.

Wir haben den MPS 801 und den Seikosha GP500A miteinander verglichen. Die Mechanik beider Drucker ist identisch und auch beim Gehäuse bestehen kaum Differenzen. Der Unterschied liegt im Verborgenen, denn der MPS 801 wurde mit den gleichen Steuerbefehlen wie der seit langem bekannte 1525 (baugleich mit Seikosha GP100VC) ausgestattet. Dazu aber später mehr. Betrachten wir zunächst den GP500A (Bild 5).

Mit einer Centronics-Schnittstelle ausgestattet, ist der GP500A nicht direkt an den C 16 anschließbar. Es wird deshalb notwendig, zusätzlich ein Interface anzuschaffen. Einziger Vorteil dieses Druckers gegenüber dem MPS 801 wäre der vorhandene deutsche Zeichensatz. Von dem kann der



Bild 5. Seikosha GP500A, der Nachfolger des GP100

```
DER SEIKOSHA GP 500A
VERUEGT UEBER EINE PAPIERBREITE
VON 21 ZENTIMETERN
ER BESITZT EINE 5X8 ZEICHENMATRIX
UND EINE DRUCKGESCHWINDIGKEIT VON 50
 ZEICHEN PRO SEKUNDE
DER GP500A IST GRAFIKFAEHIG
DOPPELTE SCHRIFTBREITE
DURCH CHR$(14)
```

(Schriftbild verkleinert)

Bild 6. Schriftprobe vom MPS 801 und GP500A



Bild 7. MPS 801 von Commodore

C 16-Besitzer aber wenig Gebrauch machen, denn das Schriftbild (Bild 6) ist eigentlich nicht ausreichend. Der GP500A kann keine Unterlängen drucken. Buchstaben wie »p« oder »y« werden immer angehoben, was einem harmonischen Textbild nicht gerade zuträglich ist. Wer ihn zum Programmieren verwenden möchte, stößt recht bald auf die Grenzen. Außer einer vergrößerten Schrift und einem Grafikmodus sind kaum Sonderfunktionen vorhanden. Der GP500A kostet 599 Mark.

Dem GP500A ähnlich ist der GP50A. Er ist ebenfalls ein Nadel-Matrixdrucker, bei dem die Papierbreite allerdings halbiert wurde. Auch er verfügt nur über eine Centronics-Schnittstelle. Das Haupteinsatzgebiet dieses Druckers wäre das eines preiswerten (398 Mark) Protokolldruckers beim Programmieren. Dazu fehlt ihm aber der Commodore-Zeichensatz. Da er diese Fähigkeit erst zusammen mit einem Interface erlangt, geht leider einiges vom Preisvorteil verloren. Die Handbücher zu den beiden Druckern sind ziemlich kurz gehalten und nicht auf das Commodore-Basic abgestimmt.

Die Problemlosen

Der MPS 801 (Bild 7) ist eine Weiterentwicklung des 1525 (baugleich mit Seikosha GP100VC, der vom GP500C abgelöst wurde). Alle Steuerzeichen und Sekundäradressen des C 16 entsprechen denen des Druckers. Der Unterschied liegt im etwas modernisierten Gehäuse und einer anderen Druckmechanik (die des Seikosha GP500A). Das Farbband wurde gegenüber dem 1525-Drucker verkleinert und direkt auf dem Druckkopf in einer kleinen Kassette untergebracht. Durch die neue Mechanik ist der MPS 801 etwas schneller als sein Vorgänger geworden, er schafft jetzt 50 Zeichen pro Sekunde gegenüber 30 Zeichen pro Sekunde beim 1525. Der MPS 801 kostet 298 Mark, bietet aber nur wenig Vorteile gegenüber dem 1525, der nur noch auf dem Gebrauchtmarkt erhältlich ist. Er eignet sich vor allem als Grafik-Drucker oder zum LISTen von Programmen. Für die Textverarbeitung gilt das gleiche wie für den Seikosha GP500A. Die Zeichendarstellung erreicht leider keine Briefqualität (Bild 6). Als Ergänzung zum MPS 801 bietet Commodore auch noch den MPS-803-Drucker an, der mit 398 Mark den gleichen technischen Standard bietet, aber teurer ist und weniger bietet. So fehlt ihm die Traktorführung, die gegen einen Aufpreis von 100 Mark zu erhalten ist.

Der leise Star

Der Star STX 80 (Bild 8 und 9) ist einer der leistungsfähigsten Drucker dieses Testes. Als Thermo-Drucker konstruiert, ist er fast nicht zu hören. Der STX 80 schafft im bidirektionalen Druck bis zu 60 Zeichen pro Sekunde. Alle Buchstaben haben Unterlänge und sogar deutsche Umlaute sind vorhanden. Schade, daß der STX 80 nur mit Spezialpapier drucken kann, denn sonst wäre er der einzige auch zur Textverarbei-

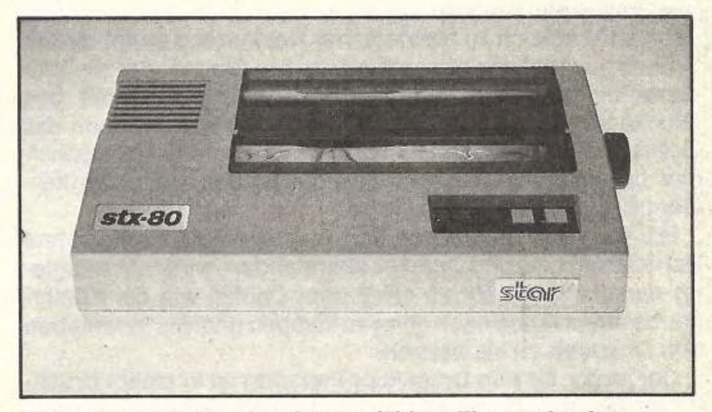


Bild 8. Star STX 80, ein leistungsfähiger Thermodrucker

```
DIE FUNKTIONEN DES STAR STX 80
DRUCKER MIT CBM INTERFACE
AUSDRUCK IN HEXADEZIMALER FORM
41 42 43 44 45 46 0D
Gross und Kleinschreibung
EIGENE GRAFIKZEICHEN

DOPPELTE BREITE
REVERSE ZEICHEN

ALLE GRAFIKZEICHEN

UND HARDCOPYS MIT SIMONS-BASIC:
```

(Schriftbild verkleinert)

Bild 9. Das Schriftbild des vielseitigen und leisen Star STX 80

tung einsetzbare Drucker. Sein Schriftbild erfüllt die Mindestanforderungen. Seine wahren Fähigkeiten zeigt der STX 80, wenn er mit dem Star-Interface an den C16 angeschlossen wird. Bild 9 zeigt die umfassenden Möglichkeiten, die dem Programmierer dann zur Verfügung stehen. Die Befehle des Interfaces erlauben sogar einwandfreie Listings in Klarschrift (Steuerzeichen werden übersetzt).

Mit einem Preis von 595 Mark ohne Interface bietet der STX 80 viel für sein Geld. Er ist der ideale Drucker für alle, die gehobene Ansprüche stellen, denen aber Nadel-Matrixdrucker zu laut sind. Sein größter Nachteil sind die relativ hohen Kosten für das Spezialpapier. Das Handbuch für den Drucker und das Interface kann als gelungene Produktbeschreibung bezeichnet werden.

Der Präsident 6313 C – das preiswerte Schwergewicht

Mit rund sieben Kilogramm stellt sich der Präsident 6313 C als ein Schwergewicht unter den Druckern vor. Aber ist er auch ein Meister seiner Klasse? Wir haben es getestet.

Der Präsident 6313 C (Bild 10) erweckt zunächst den Eindruck, als ob er aus einer anderen Welt käme. Tatsächlich ist dieser Eindruck gar nicht so falsch, denn er ist ein ostdeutsches Produkt, das versucht, mit dem westlichen Standard Schritt zu halten.

Zwar ist ansprechendes Design (er wirkt etwas klobig) nicht gerade seine Stärke, aber ein praktisches Gerät wie einen Drucker kauft man ja nicht nur als Schmuckstück für die Wohnung. Wer möglichst lange etwas von seiner Investition haben möchte, wird vielmehr Wert auf andere Attribute legen. In dieser Hinsicht hat der Präsident 6313 C allerdings einiges zu bieten. Er besitzt ein Chassis aus zwei Millimeter starkem Stahlblech, das alle anderen ebenfalls äußerst robust ausgeführten Teile aufnimmt. Dieser Eindruck wird dann noch bestärkt, wenn man den Drucker das erste Mal hochhebt, denn im Vergleich zu fernöstlicher Konkurrenz mutet dieses Schwergewicht wie ein prähistorischer Dinosaurier an. Wer dabei nicht aufpaßt, findet übrigens das Unterteil des Druckers recht unsanft auf seinen Zehen wieder, denn das Gehäuseoberteil ist erstaunlicherweise nur mit einem Scharnier befestigt. Bei diesem Gewicht ist das sicherlich kein Genuß.

Hat man das Auspacken und Aufstellen bis hierher ohne Schaden an Leib und Drucker überstanden, wird man neugierig darauf, den leicht zu öffnenden Deckel wie die Kühlerhaube eines Autos nach oben zu klappen und das Innenleben des Druckers zu inspizieren.

Der Motor für den Druckkopf-Transport ist in einem Druckgußgehäuse untergebracht, die Druckkopfführung besteht aus einem Vierkantprofil und auch sonst findet man Eisen, soweit das Auge reicht.

Der Druckkopf scheint etwas überdimensioniert, was aber sicher nicht negativ zu bewerten, sondern eher Ausdruck östlichen Technologiestandards ist.

Die Farbbandkassette hat die Maße 12 x 13 Zentimeter und läßt sich vorbildlich einfach einlegen, respektive wechseln. Die Finger behalten dabei ihre ursprüngliche Farbe, was letztendlich auch der Kleidung zugute kommt, denn Druckerfarbe gehört bekanntlich zu den hartnäckigsten Flecken.

Die Einstellung der Anschlagstärke ist, wie man nun schon fast erwartet, mechanisch einwandfrei, aber nur mit schlanken Fingern problemlos zu verstellen.

Der Antrieb des Druckwerkes erfolgt mit einem Stahlseilzug, der sogar eine Spannvorrichtung in Form einer einfachen Feder aufweist.

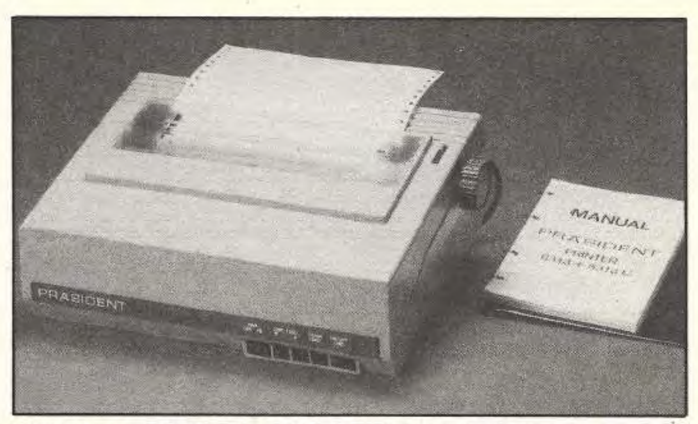


Bild 10. Der Präsident – robust, preiswert und Epson FX80-kompatibel

Präsident 6313 C
Schönschrift (NLQ)
Normalschrift
Breit
Fettschrift
Doppeldruck
NLQ+Fettschrift
NLQ+Doppelschrift
Breit/Fett
Hoch und Tief

Bild 11. Schriften, Made in DDR – der Präsident

NLQ

Bild 12. Die NLQ-Schrift in fünffacher Vergrößerung

Der Präsident 6313 C verarbeitet sowohl Endlos- als auch Rollenpapier und Einzelblätter, die allerdings nicht automatisch eingezogen werden. Letztere werden ähnlich einer Schreibmaschine von oben hinter der Gummiwalze eingeführt und dann ausgerichtet. Der notwendige Papierlöser ist unmittelbar am Drehkopf angebracht. Bei Endlos- oder Rollenpapier wird der Einführschlitz an der Rückseite des Präsident 6313 C benutzt. Sicher ungewöhnlich ist die Anordnung der Stachelwalzen auf einer Achse mit der Gummiwalze. Leider ist er so gestaltet, daß beim Abreißen des Papiers an der nicht besonders scharfen Abreißkante immer ein Teil des nächsten Blattes verlorengeht.

Geradezu fortschrittlich erscheinen dagegen die 36 Mikroschalter, die unübersehbar auf der Vorderseite des geöffneten Präsident 6313 C angebracht sind. Diese Fülle an Schaltern löst sofort emsiges Blättern im (ost)deutschen Handbuch aus. Dort findet man dann Erstaunliches: Der Präsident 6313 C ist kompatibel zum Epson-Standard, zum IBM-PC, zu Commodore-Computern und auch zu Schneider-Computern soll er passen. Sollte hier etwa doch ein Meister aller Klassen des Arbeiter- und Bauernstaates stehen? Es sei an dieser Stelle vorweggenommen: Der Präsident 6313 C zeigt, daß nicht nur im fernen, sondern auch im nahen Osten gute Drucker gebaut werden. Das Schnittstellenproblem ist durch das schon vom Epson GX-80 bekannte Modulkonzept gelöst worden. An der Rückseite des Druckers wird ein Modul ein-

geschoben, das die gesamte Elektronik für die jeweilige Schnittstelle enthält. Doch zurück zu den Schaltern. Mit ihnen wird zunächst die Wahl unter den verschiedenen Modul-Betriebsarten getroffen. Damit erhalten die restlichen Schalter völlig unterschiedliche Bedeutungen. Die Wahl aus zehn internationalen Schriftsätzen ist möglich, Pica, Elite, komprimierte- oder fette Schrift können hier fixiert werden (Bild 11). Auch die recht ansprechende NLQ-Schrift hat einen eigenen Schalter. Alle Auswahlmöglichkeiten sind auch mit dem CHR$-Befehl zu erreichen, außer Kraft zu setzen oder zu verändern. Der Präsident 6313 C ist mit 96 Zeichen pro Sekunde in der Normalschrift kein Meister der Geschwindigkeit, aber er nadelt zuverlässig seine Zeichen aufs Papier und auch vor der Grafik schreckt er nicht zurück. Im NLQ-Modus befindet er sich mit 23 Zeichen pro Sekunde in guter Gesellschaft. Die Nadeln hämmern dabei mit einer Kraft aufs Papier, die ausreichend für bis zu drei Durchschläge ist.

Empfehlenswert

Der Drucker Präsident 6313 C ist ein gutes Beispiel dafür, daß preiswerte Computertechnik nicht immer nur aus dem fernen Amerika oder Japan kommen muß. Bedenkt man, unter welchen Schwierigkeiten dieser Drucker dort entstanden sein muß, denn in der DDR gehört schon ein simpler EPROM zu den knappen Gütern, kann man die Konstrukteure nur beglückwünschen. Mit einem Preis von 798 Mark ist er der derzeit preiswerteste Drucker mit NLQ-Fähigkeit, bei dem keinerlei Abstriche an der mechanischen Festigkeit gemacht wurden. Da der Präsident 6313 C nicht gerade als ausgesprochene Schönheit gelten kann, kommt es bei ihm viel mehr auf seine elektronischen und mechanischen Fähigkeiten an – und die stimmen rundum.

Lohnt sich der Kauf eines Low-Cost-Druckers?

Der Test hat gezeigt, daß bei Low-Cost Druckern der unteren Preislagen von 300 bis 700 Mark teilweise große Abstriche an Qualität und Leistungsfähigkeit gemacht werden müssen. Keines der getesteten Geräte in diesem Bereich erfüllte alle an einen Drucker zu stellenden Anforderungen in ausreichenden Maße. Die schwierigste Hürde, die Eignung zur Textverarbeitung, haben bis auf eine Ausnahme alle Testkandidaten dieser Preisgruppe nicht überwinden können. Entweder reicht die Qualität der Schrift für heutige Ansprüche kaum aus, oder die Papierbreite, beziehungsweise Papierart, behindert eine sinnvolle Anwendung. Trotzdem haben Low-Cost-Drucker ihren Markt, sind sie doch oft die einzig erschwinglichen Alternativen. Dennoch sollte jeder überprüfen, ob er nicht doch etwas mehr anlegen kann, denn ab zirka 700 Mark gibt es heute schon Drucker, deren Schriftbild und Leistungsfähigkeit ausreichend sind, wie der im Test aufgeführte »Präsident 6313«.

Der Wiederverkaufswert eines Druckers sinkt wegen der vielen mechanischen Teile schneller als bei einem Computer, daher solte man Drucker lieber eine »Nummer größer« kaufen, denn mit steigender Programmierfähigkeit wachsen auch die Ansprüche hinsichtlich des Druckers.

(do/aw)

Info:
Brother International, Im Rosengarten 14, 6368 Bad Vilbel, Tel. (06193) 8050;
Star, Frankfurter Allee 1-3, 6236 Eschborn/Ts., Tel. (06169) 70180;
Commodore, Lyonerstr. 38, 6000 Frankfurt 71, (069) 66380;
Epson, Am Seestern 24, 4000 Düsseldorf, Tel. (0211) 595210;
Microscan, (Seikosha-Drucker), Oberseering 31, 2000 Hamburg 60, Tel. (040) 6320030.
Horst Grubert, Import + Agentur, 8110 Riegsee, Tel. (08841) 8011

Marktübersicht: Matrixdrucker

Ständig steigt das Angebot von Matrixdruckern für Heimcomputer. Ein Grund, den Druckermarkt zu durchleuchten und Ihnen eine Einkaufshilfe zu geben.

Immer mehr Besitzer von Heimcomputern entscheiden sich zum Kauf eines Matrixdruckers. Die Möglichkeiten dieser Drucker gehen vom einfachen Listingsdruck bis zur Korrespondenz in Schönschrift. Sie müssen also entscheiden, wozu Sie den Drucker verwenden wollen. Zum Druck von Listings gibt es inzwischen sehr günstige Geräte, deren Schriftbilder durchaus akzeptabel sind. Achten Sie darauf, daß der Drucker zu diesem Zweck auch die Grafikzeichen des C 16 beherrscht. Der Anschluß des Druckers kann auf verschiedene Arten erfolgen. Commodore-Drucker werden über den seriellen Bus des C16 angeschlossen. Centronics-kompatible Drucker haben einen parallelen 8 Bit breiten Eingang. Ein spezielles Interface ist also zusätzlich nötig. Der Preis dafür liegt zwischen 50 und 300 Mark.

Wollen Sie HiRes-Bilder (HiRes = hochauflösende Grafik) des C64 zu Papier bringen, muß der Drucker grafikfähig sein. Das heißt, seine Nadeln müssen einzeln ansteuerbar sein.

Zum Druck von langen Listings und viel Korrespondenz ist eine hohe Druckgeschwindigkeit von Vorteil. 30 Zeichen pro Sekunde, das hört sich beim Studieren von Katalogen recht schnell an, wird aber schon beim zweiten Ausdruck zur Langweilerei.

Die Geräuschentwicklung des Druckkopfes sollte nicht außer acht gelassen werden. 70 dB(A) um Mitternacht sind für Mietwohnungen einfach zu laut.

Wie Sie sehen, Drucker ist nicht gleich Drucker. Ein hoher Qualitätsstandard muß zwar bezahlt werden, aber es gibt durchaus erschwingliche Drucker, die über eine hohe Druckqualität in den verschiedensten Schriftarten verfügen.

(do/hm)

Worauf sollten Sie beim Kauf achten?

Wenn Sie folgende Punkte beim Kauf eines Druckers berücksichtigen oder zumindest in Ihre Überlegung einbeziehen, können Sie nicht viel falsch machen.

Schriftbild:
* Der Drucker sollte möglichst Unterlängen auch als Unterlängen drucken.
* Auch deutsche Umlaute dürfen für ihn keine Hürde sein.
* Der Commodore-Zeichensatz erleichtert das Lesen von Listings.

Technischer Standard:
* Anschlußfähig für den C64.
* Die Steuerung muß über DIL-Schalter und softwaremäßig als »PRINT CHR$(n)«-Anweisung ausführbar sein.
* Er sollte Einzelblatt und Endlospapier verarbeiten können.
* Ein stabiles und handliches Gehäuse wäre wünschenswert.
* Farbband und DIL-Schalter sollten leicht zugänglich sein.

Bedienung und Handhabung:
* Einfaches Einlegen des Papiers.
* Eine Druckbreite von mindestens 80 Zeichen pro Zeile ist empfehlenswert.
* Er sollte Normalpapier verarbeiten, denn Thermopapier ist ziemlich teuer.

a) Anbieter b) Hersteller	Modell	Anzahl der Nadeln	Zeichenmatrix	Grafikfähig	Grafische Auflösung Punkte/Zoll	Anzahl Zeichensätze/ Schriftarten	Druckgeschwindigkeit Zeichen pro Sekunde	a) Papierart: endlos (1), Einzelblatt (2), Rolle (3) b) Antrieb: Walze (1), Traktor (2)	Papierbreite	a) Durchschläge inkl. Original b) max. Druckbreite in Zeichen	Lautstärke in dB (A)	Druckpuffer (in KByte)	Abmessungen (HxBxT) in cm	Schnittstellen: RS232 = 1, Centronics = 2, C64/VC20/C16 = 3	a) Druckerkabel im Lieferumfang enthalten b) anschlußfertig an C 64 c) Mitgeliefertes Interface	Empfohlenes Interface	Preis inkl. MwSt.
a) **Commodore**	MPS 801	7	6x7	ja	7 Punkte/ Spalte	1	50	a) 1 b) 2	min. 11,43; max. 25,4	a) 3 b) —	k. A.	0	13,6x42,5x23,5	3	a) ja b) ja c) nein	—	298,-
b) Commodore	MPS 803	7	6x7	ja	k. A.	k. A.	60	a) 1 b) 2	max. 25,4	a) 3 b) k. A.	k. A.	0	7x33x19	3	a) ja b) ja c) nein	—	398,-
a) **C. ITOH Electronics GmbH** b) C. Itoh	Riteman C plus	9	9x9	ja	60, 120	4/—	105 bei 10 cpi	a) 1, 2 b) 1, [illegible]	min. 10,4; max. 25	a) 2 b) 80	<60	2	7,3x35,6x26,8	3	a) ja b) ja c) ja	—	998,—
a) **Horst Grubert, Import + Agentur** b)	Präsident 6313	9	11x9	ja	k.A.	4/6	100	a) 1, 2, 3 b)	min: 85 max. 25,2	a) 2 b) k.A.	< 58,5	k.A.	13x37x28	3	a) ja b) ja c) -	—	798,-
a) **Macrotron** b) Macrotron	Speedy 100-80	9	9x8	ja	72—144	3	100	a) 1, 2 b) 1, 2	min. 10; max. 25,5	a) 4 b) 142	58	2-4	14,0x38,0x29,5	1, 2, 3	a) ja b) ja c) ja	—	1007.—
a) **Melchers** b) Shinwa	CPA-80C	9	7x8/8x9	ja	640x8,9/ 1280x8	10/6	100	a) 1, 2, 3 b) 1, 2	min. 10; max. 25,4	a) 2 b) 142	60	0,5	12,5x38,4x31,5	3	a) ja b) ja c) -	—	748,—
	CP-80X	9	8x8/8x9	ja	640x8/ 1280x8	8/4	80	a) 1, 2, 3 b) 1, 2	min. 10; max. 25,4	a) 3 b) 142	60	0,5	12,5x37,7x29,5	2, 3, 1 als Option	a) ja b) ja c) -	—	838,—
a) **Microscan** b) Seikosha	GP-500 VC	7	5x7	ja	480 Punkte/ Zeile	CBM-ASCII-Code	50	a) 1 b) 2	min. 11,4; max. 25,4	a) 2 b) 80	<60	1 Zeile	11,4x44,7x31,5	3	a) ja b) ja c) ja	—	399,—
	GP-550 AVC	9	5x8-24x16	ja	960	8/16 CBM-ASCII	40-96	a) 1, 2 b) 1, 2	min. 11,4; max. 25,4	a) 2 b) 139	<60	1 Zeile	11,3x42x30,5	2, 3	a) ja b) ja c) ja	—	599,—
a) **Mirwald Electronic** b) Mirwald-BMC	BX 80	9	8x9	ja	85, 170	8/18	80	a) 1, 2, 3 b) 1, 2	min. 10,16; max. 25,4	a) 4 b) 142	—	142 Bytes	12,5x37,7x29,5	1, 2, 3, IEEE488	a) ja b) ja c) Aufpreis	—	748,—
	BX 100	9	9x9	ja	60, 120, 80, 72, 90, 240	8/32	100	a) 1, 2, 3 b) 1, 2	min. 10; max. 25,4	a) 2 b) 132	<59	132 Zeich.	12,0x40,0x30,0	1, 2, 3, IEEE488	a) ja b) ja c) Aufpreis	—	998,—
a) **Star Europe** b) Star	STX 80	5x9	5x9 6x6	ja	60	1/1	60	2,3	22	a) — b) —	k. A.	1 Zeile	35,2x19x10	2	a) nein b) nein c) ja	Star C64 Interface	595,—

C 16 und Diskette

Geschwindigkeit ist Trumpf. Deshalb haben wir für Sie die entscheidenden Unterschiede zwischen Datasette und der schnelleren Diskettenstation zusammengestellt

Um die Effektivität des C 16 zu steigern, ist es notwendig, sich zum Computer und der eventuell schon vorhandenen Datasette auch noch ein Diskettenlaufwerk anzuschaffen.

Passend zum C 16 wäre die 1541 von Commodore, deren Preis etwa um 500 Mark liegen dürfte, zuzüglich einiger Disketten, zwischen 5 und 10 Mark das Stück. Mit dieser Zusammenstellung steht dem C 16 die gesamte Programmwelt auf Diskette zur Verfügung, solange ein RAM von 16 KByte ausreichend ist. (Bild 1)

Die Vorteile

Welche Vorteile bietet nun die Diskette gegenüber einer Datasette?

1. schnellere Zugriffzeiten
2. wahlfreier Zugriff auf alle Daten
3. größere Datenmengen
4. Inhaltsverzeichnis
5. Erweitern und Kürzen von Datenblöcken
6. direkter Zugriff auf Datenblöcke
7. Suchroutinen für das Inhaltsverzeichnis
8. Aufräumroutinen

Geschwindigkeit ist keine Hexerei

Die um den Faktor 8 schnellere Busgeschwindigkeit erspart dem Benutzer lange Wartezeiten, die er von seiner Datasette her gewohnt ist. Somit kann der Benutzer auch längere Programme oder größere Datenmengen schnell laden oder speichern. Die Geschwindigkeitssteigerung wird einerseits durch die höhere Übertragungsrate (gemessen in Bits pro Sekunde) von 1200 gegenüber 300 bit/s bei der Datasette erreicht. Des weiteren entfällt bei der Diskettenstation die doppelte Aufzeichnung, die zur Sicherheit der einzelnen Datenblöcke dient. (Ein Datenblock ist eine 256 Byte lange Informationseinheit. Ein Programm oder eine Datei besteht in der Regel aus mehreren dieser Datenblöcke).

Wahlfreier Zugriff

Ein weiterer Vorteil ist der ständige und wahlfreie Zugriff auf alle Programme und Dateien, die auf der Diskette gespeichert sind. Damit erspart man sich, das Band ständig vom Anfang her absuchen zu müssen, bis das entsprechende Programm oder eine Datei gefunden ist. Über eine besondere Befehlsform (Relative Datei) ist es möglich, auch auf Daten innerhalb einer Datei zuzugreifen, gegenüber dem Einlesen einer ganzen Datei (serielle Datei) und deren späterer Zerlegung. Zum Beispiel kann aus der Datenmenge »Otto Meier 24 Jahre« auf das Alter »24« direkt zugegriffen werden. Somit kann ein Sortierprogramm nach den Altersangaben realisiert werden.

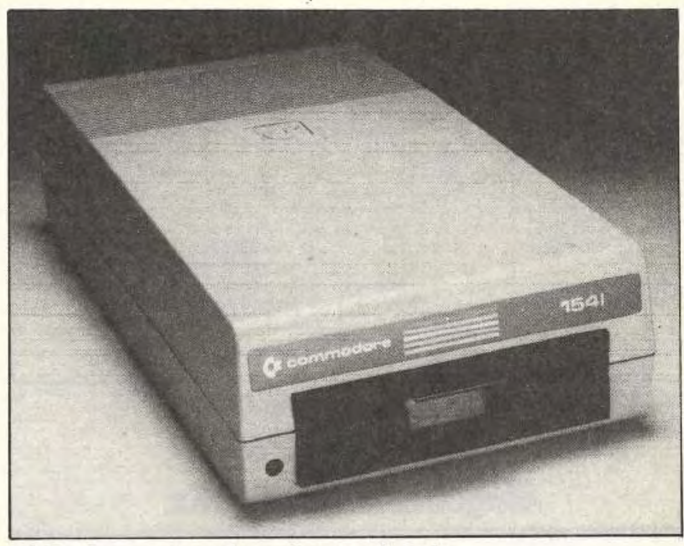


Bild 1. Commodore-Floppy-Laufwerk 1541

Um die gleiche Datenmenge, die auf einer Diskettenseite (174848) Byte gespeichert werden können, mit einer Datasette auf Kassetten zu speichern, würde man zirka vier Kassetten benötigen. Davon abgesehen, benötigen vier Kassetten erheblich mehr Platz als eine »halbe« Diskette.

Eine Diskette ist in mehrere Spuren und Sektoren unterteilt, auf denen die Daten gespeichert sind (Spuren sind die Speicherzeilen, die ähnlich wie auf einer Schallplatte konzentrisch um den Mittelpunkt angelegt sind. Diese sind des weiteren unterteilt in Blöcke oder Sektoren, in denen jeweils 256 Byte in Datenblöcken gespeichert sind (Bild 2, Grafik)).

Direktanwahl per Menü

Auf der Spur 18 befindet sich das Inhaltsverzeichnis, das bei einer für die 1541 formatierten Diskette (Formatierung ist die Einteilung einer werkfrischen Diskette in die notwendigen Spuren und Sektoren vor dem ersten Gebrauch) das sogenannte »Directory« beherbergt. In diesem sind die Benennung sowie die Ortsangabe vermerkt, wo die Daten auf der Diskette gespeichert sind. Dafür wird zusätzlich noch vermerkt, ob es sich um eine Programmdatei ».prg« (lauffähiges Programm), serielle Datei ».seq« (gesamte Datenlänge einlesend) oder relative Datei ».rel« (ausschnittweise lesend) handelt.

Mit Hilfe des Directorys läßt sich eine umfangreiche und übersichtliche Programmbibliothek aufbauen, die es ermöglicht, einen wahlfreien Zugriff auf alle Dateien zu realisieren.

Ein weiteres Leistungsmerkmal stellt der direkte Zugriff dar; er ermöglicht es, gezielt auf bestimmte Spuren und Sektoren zuzugreifen, um begrenzte Datenmengen zu lesen, zu schreiben oder zu ergänzen (siehe Listing).

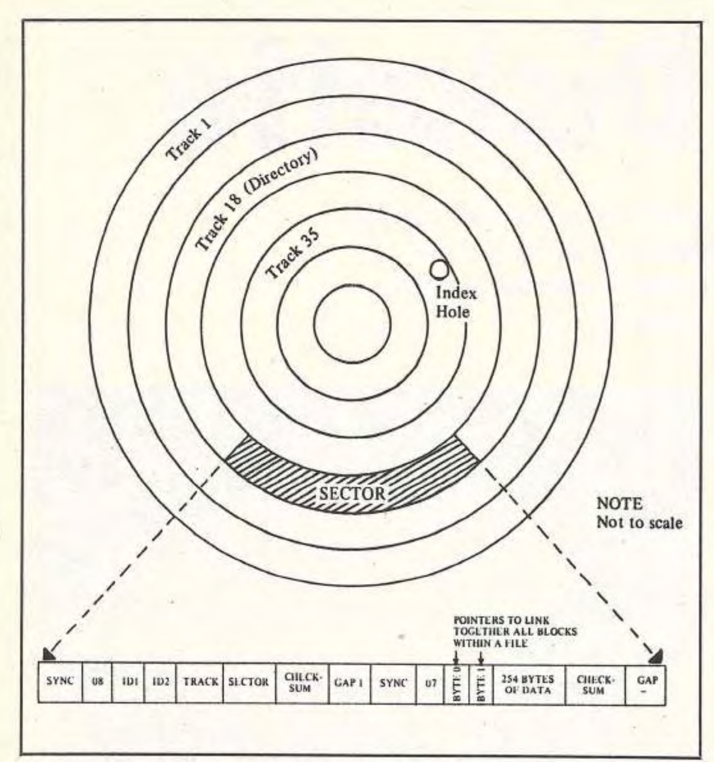


Bild 2. Technisches Aufzeichnungsformat

Eingebaute Intelligenz

Mit Hilfe der festeingebauten Firm-Software im sogenannten DOS (Disk Operating System) ist die Diskettenstation in der Lage, Programme zu überschreiben, zu löschen, zu verlängern, ohne andere Dateien oder Programme zu überschreiben oder zu zerstören, wie es bei der Datasette unvermeidbar ist. Dazu schaut das DOS im Directory nach, ob und wo noch Platz ist, und legt die Dateien/Programme entsprechend dort ab.

Sortierroutine fest eingebaut

Mit einem Befehl zum Abfragen des Directorys ist es möglich, Dateinamen mit gleichen Anfangsbuchstaben, Anfangssilben oder Wortendungen auflisten zu lassen. Damit läßt sich eine Dateiauswahl schnell und komfortabel erstellen. Diese DOS-gebundenen Routinen können von einem versierten Programmierer in seine Software voll eingebunden werden. Ebenso ist die gesamte Diskettenstation auch per Maschinensprache ansprechbar und steuerfähig. Es ist möglich, eigene Routinen in das Hauptprogramm einzuketten und mitlaufen zu lassen, ausgenommen von 2 KByte RAM, die nur unter Maschinensprache ansprechbar sind.

Diskstation mit eingebauter Müllabfuhr

Nicht mehr gebrauchte Dateien werden vom DOS mit einem ».del« = gelöscht (scratch) im Directory gekennzeichnet. Das DOS behandelt die gelöschten Dateien so, daß beim Erstellen eines Inhaltsverzeichnisses dieser Eintrag überlesen wird.

Auf Anforderung führt das DOS eine Validate-Routine durch, die alle gelöschten Dateien aus dem Directory verbannt und deren Platz wieder als frei kennzeichnet.

```
0
1     "PROGRAMM-BANK"    PRG
32    "BANK.EPSON"       PRG
13    "BANK.SORT"        PRG
1     "BANK.DIR"         PRG
2     "WRITE-PROTECT"    PRG
303   "BANK.REL"         REL
13    "BANK.INDEX"       SEQ
1     "MASTER-SLAVE"     USR
298 BLOCKS FREE.

READY.
```

Listing des Directorys

Reparaturwerkstatt eingebaut

Gelöschte Dateien lassen sich durch ein entsprechendes Programm oder eine spezielle Befehlsfolge wieder voll restaurieren (nur die mit scratch gelöschten Dateien), indem die Kennung wieder mit ».prg« = Programmdatei, ».seq« = Sequentielle Datei, ».rel« = Relative Datei, ».usr« = User Datei vervollständigt wird, so daß die Datei wieder voll verarbeitungsfähig ist.

Die Empfehlung

Anwender, die häufiger mit einem C 16 arbeiten möchten, sollten sich unbedingt eine Diskettenstation zulegen. Diejenigen, die mit dem Computer größere Dateimengen verwalten wollen, kommen ohne die Möglichkeit des Direktzugriffes fast nicht mehr aus, wenn sie nicht mit den umständlichen sequentiellen Dateien arbeiten wollen. Des weiteren ersetzt die Diskstation den fehlenden RAM-Bereich. Deshalb empfehlen wir, wenn es der Geldbeutel erlaubt, sich eine Diskettenstation zum C 16 zuzulegen.

(do)





Bücher zum C 16 und VC 20

Das Buchangebot für den C 16 und VC 20 ist leider bei weitem nicht so umfangreich wie beispielsweise beim C 64. Speziell für den C 16 gibt es neben einigen Büchern, die nicht mehr Information als das Original-Handbuch enthalten, kaum wirklich informationsreiche Literatur. Die beiden hier besprochenen Bücher zum C 16 können dem Computer-Neuling zur Lektüre empfohlen werden, denn sie enthalten neben einem Überblick über die Datenverarbeitung allgemein auch vieles, was zum Erlernen von Basic mit dem C 16 nützlich ist.

Zum VC 20 finden Sie eine Auswahl empfehlenswerter Bücher, die sich in erster Linie mit fortgeschrittener Programmierung und Tips & Tricks beschäftigen.

Basic-Wegweiser für den Commodore 116, Commodore C 16 und Commodore Plus/4

Dieses Buch mit dem Untertitel »Datenverarbeitung mit Basic 3.5« zeigt dem Einsteiger in die Welt der Computerprogrammierung anschaulich die vielfältigen Einsatzmöglichkeiten der Computerserie rund um den C 16. Das 300 Seiten umfassende Werk läßt sich in drei große Abschnitte gliedern, die vom Umfang her jeweils etwa ein Drittel des Buches beanspruchen.

Der erste Abschnitt befaßt sich mit den Grundlagen der Datenverarbeitung ganz allgemein, leider ohne speziell auf den C 16 einzugehen.

Diejenigen Leser, die sich konkret für die Bedienung und Programmierung ihres C 16 interessieren, aber an den theoretischen Grundlagen der Computer-Technologie weniger Interesse haben, können die ersten hundert Seiten getrost überschlagen, denn erst im zweiten Abschnitt geht es konkret um den C 16, das dann allerdings sehr ausführlich und dank vieler Beispiele auch gut verständlich. Man erfährt einiges über den allgemeinen Umgang mit seinem Computer und über den Aufbau von Basic-Programmen. Alle Befehle des Basic 3.5 werden anhand von kurzen Beispielen vorgestellt und erläutert.

Der dritte Abschnitt des Buches enthält schließlich einen kompletten Basic-Programmierkurs, speziell abgestimmt auf den C 16.

Erfreulicherweise erschöpft sich dieser Kurs nicht mit einer nochmaligen Darstellung der Basic-Befehle, sondern stellt auch die grundlegenden Methode der Planung und Entwicklung von Software vor. Spezielle Kapitel befassen sich hier mit Text- und Tabellenverarbeitung, mit Suchen und Sortieren von Daten, mit maschinennaher Programmierung sowie mit Dateiverarbeitung und Grafik-Programmierung.

Insgesamt ein für den an der anwendungsorientierten Basic-Programmierung interessierten C 16-Besitzer sehr empfehlenswertes Buch. Leider findet man in dem gesamten 300 Seiten umfassenden Werk weder Angaben über die Speicherbelegung des C 16 (Zero-Page, Betriebssystem etc.) noch überhaupt Informationen über das Betriebssystem und dessen Nutzung per Maschinensprache oder Basic. Auch wer Spiele für seinen Computer entwerfen möchte, ist mit diesem Buch leider nicht richtig bedient.

(Christoph Sauer/ev)

Info: Dr. Ekkehard Kaier: Basic Wegweiser für den Commodore 116, Commodore 16 und Commodore Plus 4, Vieweg-Verlag, 300 Seiten, ISBN 3-528-04337-7, Preis 48 Mark.

Commodore 16 mit 116

Wer zu denjenigen gehört, die sich einen C 16 zugelegt haben, wird sich über jedes Buch freuen, das sich mit seinem Computer beschäftigt; denn bisher sind nur einige wenige Werke über dieses Gerät erschienen. Dieses Buch über den »Einsteigercomputer mit Aufsteigerqualitäten« (Untertitel) gehört zu den ersten Büchern zum C 16.

Die ersten zwei Kapitel enthalten Allgemeines über Datenverarbeitung und »Software von der Stange«, sind also nicht auf den C 16 ausgerichtet. Erst im dritten und vierten Abschnitt stellt man überhaupt fest, daß dies ein C 16-Buch ist, denn hier geht der Autor auf die Tastaturbedienung, Diskettenbefehle und die Toolkitfunktionen (wie zum Beispiel RENUMBER) ein.

Das neunte Kapitel ist der unbestrittene Glanzpunkt des ganzen Buches. Die Beschreibung der einzelnen Grafikbefehle des Basic 3.5 ist sehr gut gelungen. Zahlreiche Beispiele ermöglichen auch dem Neuling die schnelle Einarbeitung in die Welt der Computergrafik. Insgesamt ein für den Einsteiger durchaus empfehlenswertes Buch, das von den allgemeinen Grundlagen der Datenverarbeitung bis zur Grafikprogrammierung mit dem C 16 führt. Für den schon mit etwas Computerwissen vorbelasteten Umsteiger, der vielleicht schon mit einem VC 20 programmiert hatte, ist das Buch dagegen nur bedingt geeignet: Die Erklärungen bezüglich des Basic 2.0 (also der »alten« Befehle) und des neuen 3.5 Basic stehen in keinem richtigen Verhältnis zueinander. So kann man ruhigen Gewissens behaupten, daß teilweise sogar das Handbuch ausführlicher ist. Wirklich gut gelungen und empfehlenswert ist dagegen die Beschreibung der neuen Grafik-Befehle.

(Christoph Sauer/ev)

Info: Hans Riedl, Commodore C 16 mit 116: Der Einsteigercomputer mit Aufsteigerqualitäten, Kiehl Verlag 1985, 160 Seiten, ISBN 3-470-80481-8, 32 Mark.

VC 20 Intern

Obwohl das Haus Data Becker in den letzten Jahren den Computer-Literaturmarkt mit einer Vielzahl von Werken überschwemmt hat, gehört eines der ersten zu den besten. »VC 20 Intern« war zu einer Zeit, als kaum jemand den Commodore kannte (beziehungsweise kaufte, weil er zu teuer war), das einzige Buch, das den mit seinem Handbuch alleine gelassenen VC 20-Besitzer von seinem Informationsdefizit befreien konnte.

Ähnlich einem Systemhandbuch aufgebaut, soll »VC 20 Intern« durch gute Beschreibung der Hardware und ihrer Programmierung ein ständiger Begleiter beim Programmieren sein. Diesen Anspruch unterstreicht das kommentierte ROM-Listing (in hexadezimaler Form) am Ende des Buches. Die wichtigsten Kapitel dieses Buches sind:

- Hardware (CPU, Speicherbelegungspläne, User-Port, Expansion-Port)
- Tonprogrammierung
- Grafik (Der VIC, Registerbeschreibung, Betriebsarten, Schnittstellen zum Prozessor)
- Ein/Ausgabebausteine (Register-Plan, Ports, Timer, die VIAs, Joystickprogrammierung)
- Der Basic-Interpreter (Erweiterung des Basic, Monitor-Programm, wichtige Kernal-Adressen, RS232, serieller Bus)
- ROM-Listing

Man darf ruhigen Gewissens behaupten: Dieses Standardwerk zum VC 20 ist für den fortgeschrittenen Programmierer nahezu unverzichtbar.

Info: Angershausen, Brückmann, Englisch, VC 20 Intern, Data Becker, 1983, 200 Seiten, Preis: 39 Mark

VC20-Anwenderhandbuch

Ein Buch über den VC20, das gleichermaßen den Anfänger wie den fortgeschrittenen Anwender anspricht. Neben einer ausführlichen und leicht verständlichen Einführung in die Basic-Programmierung findet der Anfänger hier auch eine Fülle von Tips und Tricks, um die Hard- und Softwareeigenschaften des VC20 optimal für eigene Programme zu nutzen. Ein ganzes Kapitel beschäftigt sich mit den Möglichkeiten zur Steuerung von Spielen mittels Joystick, Paddle oder Tastatur. Ein anderer Abschnitt beschreibt Entwurf und Realisierung benutzerfreundlicher Inputroutinen.

Ist man mit den Grundlagen seines VC20 einigermaßen vertraut, dann bieten die Kapitel über die Erzeugung von Grafiken und Tönen einen tieferen Einblick in das Innenleben des Volkscomputers. Erklärt und anhand von Beispielprogrammen demonstriert werden die Erzeugung selbstdefinierter Zeichen und – als Anwendung hiervon – die Programmierung der hochauflösenden Grafik und des Vielfarben-Modus. Sehr ausführlich dargestellt ist auch die Realisierung spezieller Toneffekte wie Schwebungen, Vibrato oder Tremolo. Bemerkenswert ist die recht umfangreiche Behandlung der wichtigsten Peripheriegeräte. Der Bogen spannt sich hier vom Farbbandwechsel beim VC 1515-Grafikdrucker bis zum Direktzugriff auf einzelne Sektoren bei der 1540-Floppy.

Zahlreiche Tabellen, Diagramme und Fotos tragen einiges zum guten Verständnis des dargebotenen Stoffes bei. Ein umfangreicher Anhang enthält die Speicherbelegung des VC20, die Funktionen des 6560-Video-Chips, alle Basic-Statements und Funktionen (auch für Diskettenbetrieb) und andere Tabellen.

Nach näheren Informationen über die verschiedenen Speicher- und Spracherweiterungsmodule sucht man im VC20-Anwenderhandbuch jedoch vergebens. Ebenfalls nicht behandelt wird der Zugriff auf die Maschinenebene.

Alles in allem handelt es sich bei dem Buch jedoch um eine recht gute Einführung für den Anfänger, aber auch der fortgeschrittene Benutzer wird hier wahrscheinlich noch einiges an neuem Wissen finden. In jedem Fall ist das Buch als umfangreiches Nachschlagewerk zum VC20-System zu empfehlen. (ev)

Info: John Heilborn / Ran Talbott: VC 20 Anwenderhandbuch, 388 Seiten, McGraw-Hill Book Company, Hamburg 1983, ISBN 3-89028-004-8, 32 Mark.

Das Kassettenbuch

Das Kassettenbuch von Data Becker beschreibt in aller Ausführlichkeit (als Ergänzung zum Commodore-Handbuch) das Zusammenarbeiten zwischen Computer und dem Datenrecorder (Datasette). Da ist zunächst die Beschreibung aller Befehle, die mit dem Zugriff des VC20 oder C64 auf die auf Band abgespeicherten Daten und Programme zu tun haben. Dies bezieht sich sowohl auf Basic als auch auf die entsprechenden Kernalroutinen, die das Arbeiten mit Maschinenprogrammen ermöglichen. Natürlich fehlt es auch nicht an guten Ratschlägen, die sich auf den Recorder selbst beziehen, wie zum Beispiel die Pflege der Andruckrollen oder das Nachjustieren des Schreib-/Lesekopfes. Auch eine Mithörkontrolle und ein Selbstbau-Kassetteninterface werden, wenn auch etwas kurz, beschrieben.

Der zweite Teil des Buches bezieht sich auf ein Fasttape-Programm (ein ähnliches ist im 64'er, Ausgabe 12 erschienen), mit dem es möglich sein soll, Programme und auch Daten 10- bis 20mal schneller abzuspeichern. Für den VC20 (und dies betrifft viele Käufer dieses Buches) ist eine Programmversion abgedruckt, die nur bei einem voll ausgebauten (28 KByte) Speicher funktioniert. Um die Routinen in einem andern Speicherbereich laufen zu lassen, müssen sie vom Leser (!) umgeschrieben werden.

Auf diese Routinen aufbauend werden dann einige andere Programme wie beispielsweise ein Kassettenverzeichnis oder ein Backup-Programm entwickelt. Diese Routinen sind alle ausführlich beschrieben und kommentiert. Was den Profi jedoch enttäuschen wird, ist das fast gänzliche Fehlen einer Beschreibung der im Betriebssystem verankerten Kassettenroutinen. Auch eine in diesem Zusammenhang notwendige Beschreibung der Ein-/Ausgabebausteine fehlt. Dafür findet man einen Hinweis auf andere Data Becker-Bücher – eine sicherlich nicht sehr befriedigende Situation.

Auch bezüglich des Aufzeichnungsformats sind die Ausführungen des Autors sehr knapp gehalten und obendrein scheinbar direkt den »Tips & Tricks«-Handbüchern des gleichen Verlages entnommen.

Wegen der gut lesbaren und informativen Einführung in das Arbeiten mit der Datasette, ist das Kassettenbuch mit seinen zahlreichen Programmen trotz einiger Schwächen eine rentable und empfehlenswerte Anschaffung für den fortgeschrittenen Einsteiger.

(Christoph Sauer/ev)

Info: Dirk Paulissen, Das Cassettenbuch zu C64 und VC20, Data Becker 1984, ISBN 3-89011-030-4, 192 Seiten, 29 Mark.

Die Floppy 1541

Hier plaudert ein Profi aus seiner Trickkiste. Denn bei diesem Floppy-Buch wurde der Schwerpunkt auf die Themen gelegt, bei denen andere aufhören. Und so erfährt man dann in lockerem Stil, was in dem Diskettenlaufwerk wirklich abläuft und wie man es geschickt manipulieren kann. Kein Blatt vor den Mund genommen wird, wenn es um Software-Schutzmethoden geht: Vielleicht ein Ärgernis für den einen oder anderen Softwarehersteller, aber nützlich für den ambitionierten Anwender, der seine selbstgeschriebene Software kopiersicher machen will. Dennoch ist das vorliegende Buch nicht nur für Profis geeignet: Wer bisher nur die Befehle LOAD und SAVE mit seiner 1541 in Verbindung bringen konnte, der erfährt hier, wie man sequentielle, relative und Direktzugriffs-Dateien realisieren und ausnutzen kann.

Einige der weiteren Themen: Fehler im Commodore-DOS werden schonungslos offengelegt. Der serielle Bus wird unter die Lupe genommen und nach Hypra-Load-Manier beschleunigt. Methoden zur Rettung von verlorengegangenen Daten und fehlerhaften Blöcken werden vorgestellt. Und dies sind längst nicht alle der angesprochenen Bereiche.

Das Allerbeste an diesem Buch ist allerdings das dokumentierte Listing des 1541-ROMs. Es ist fast zu schön um wahr zu sein, wie gründlich die Dokumentation vorgenommen wurde, denn sie läßt sich fast schon wie ein zusammenhängender Text lesen. Praktisch jeder einzelne Maschinenbefehl wurde mit einem erläuternden Text versehen, weiter gibt es zu jeder der rund 400 Einzelroutinen des DOS einen kleinen einleitenden Text, dem dann die ausführliche Dokumentation neben dem Assembler-Listing folgt.

Die Dokumentation, die fast die Hälfte des Buches in Anspruch nimmt, wird von einer ebenso ausführlichen RAM-Belegung ergänzt. Mehrere nützliche Programme und ein ausführliches Stichwortverzeichnis runden das äußerst positive Gesamtbild ab. In Vorbereitung ist eine Diskette mit allen abgedruckten und einigen zusätzlichen Programmen. Das eindeutige Urteil: Ein Floppy-Buch, wie man es sich besser kaum vorstellen kann. Von seiner Qualität her gesehen hat es gute Chancen, das Standardwerk über die 1541 zu werden.

(Boris Schneider/ev)

Info: Karsten Schramm, Die Floppy 1541, Markt & Technik 1985, 430 Seiten, ISBN 3-89090-098-4, 49 Mark.

Schnelle Spielegrafik beim C 16

Diese Einführung in die »Innereien« des C16 zeigt, wie schnelle, anspruchsvolle (Spiele-) Grafiken mit einem Zehntel des bisher nötigen Speicherbedarfs realisiert werden können.

Inzwischen hat es sich unter C16-Besitzern wohl schon herumgesprochen, daß es Schwierigkeiten bei der Kombination von Spielprogrammen mit der hochauflösenden Grafik gibt. Hier soll nun eine alternative Möglichkeit zur Grafikerzeugung aufgezeigt werden, die besonders für die Spieleprogrammierung geeignet ist. Die Rede ist von einem selbstdefinierten Zeichensatz als Ersatz für Hochauflösung und Shapes, der nur 1 KByte Speicher (statt 10 KByte bei Hochauflösung) belegt.

Auch der C16 hält, wie seine Vorgänger, einen Farb- und einen Videospeicher für die Bildschirminformationen bereit. Dieser liegt im Speicherbereich zwischen $0800 (2048) und $0FFF (4095), wobei das erste KByte dem Farbspeicher zugeordnet ist. In der Struktur der einzelnen Bits innerhalb einer Farbspeicherstelle hat sich jedoch – um wieder die anderen beiden Computer (C 64 und VC 20) zum Vergleich heranzuziehen – etwas geändert. Die untersten 4 Bit jeder Speicherstelle geben nach wie vor die jeweilige Farbe an der dazugehörigen Bildschirmspeicherposition wieder. Zusätzlich ist es jetzt beim C16 möglich, auch den Helligkeitswert jedes Zeichens individuell zu bestimmen. Dieser sogenannten Luminanz sind die Bits 4-6 (also 7 Helligkeitsstufen) zugeordnet. Bit 7 entscheidet schließlich noch darüber, ob das Zeichen normal oder blinkend dargestellt wird.

Reverse Zeichen werden im Gegensatz zu VC20 und C64 hardwaremäßig (durch Setzen des Bit 7 in der entsprechenden Videospeicherstelle) vom TED erzeugt. Somit benötigt der C16 nur noch 2 KByte Speicherplatz für die Zeicheninformationen im ROM.

Natürlich ist es auch denkbar, die Daten in den RAM-Speicher zu übertragen, um dort Änderungen am Zeichensatz vorzunehmen. Dies geschieht prinzipiell genauso wie beim VC 20 oder C 64. Man muß lediglich die neue Adresse des Zeichensatzes im Register 14 ($FF13, 65299) vermerken und auf RAM-Zugriff schalten (Bit 2 in Register 13). Das ist in der Praxis alles viel einfacher, als es im ersten Moment den Anschein hat. Dazu möchte ich zunächst eine kurze Beschreibung des beim C16 verwendeten »Bankswitching« vorausschicken: Der C16 kann hardwaremäßig 128 KByte Speicher verwalten; 64 KByte ROM und nocheinmal soviel RAM-Speicher. Da die Adreßleitungen nach wie vor aber nur für 64 KByte Gesamtkapazität ausgelegt sind, bedient man sich eines Tricks – dem Bankswitching. Das bedeutet, es wird je nach Bedarf zwischen 64 KByte ROM und RAM hin- und hergeschaltet. Dazu ist sowohl die CPU als auch der TED-Chip in der Lage. Ob er auf den Zeichensatz im ROM oder auf einen im RAM zugreifen soll, entscheidet ein Registerbit ($FF13 Bit 2). Somit ist genau dieses bei der Umschaltung auf den RAM-Zeichensatz zu ändern.

Zum besseren Verständnis zunächst ein kleines Beispiel. Dazu verwenden wir den eingebauten Monitor des C16. Als erstes wird der Zeichensatz aus dem ROM – er liegt bei Adresse $D000 – mittels des Transferbefehls (T D000 D800 1000) in den normalen Speicher kopiert. Danach ändern wir, damit man die beiden (im Moment noch identischen) Zeichensätze unterscheiden kann, das erste Zeichen (es ist der »Klammeraffe« @ in Adresse $1000 bis $1007. Dazu kann man ihn beispielsweise abwechselnd mit $00 und $FF füllen.

Als nächstes müssen die Registerinhalte so abgeändert werden, daß der TED nur noch auf unseren eigenen Zeichensatz zugreift. Dazu wird zunächst in Register 13 ($FF12) Bit 2 ausgeschaltet, damit von nun an auf RAM zugegriffen wird. In unserem Beispiel ersetzt man derzeitigen Wert $C4 durch $C0. Danach ist noch die Zugriffsadresse für den Zeichengenerator von ursprünglich $D000 auf $1000 (der neuen Adresse) zu ändern; dazu wird das High-Byte der Zugriffsadresse in Bit 4-7 dieses Registers eingetragen (statt $D1 jetzt also $11). Nun kann man anhand des Klammeraffen-Zeichens kontrollieren, ob man den richtigen Zeichensatz vor sich hat.

Ein Zeicheneditor

Mit Hilfe des Programms »Matrix-Editor« (Listing) kann man nun komfortabel am Zeichensatz »herumpfuschen«. Dazu kurz noch einige Erläuterungen. Das vorliegende Basic-Programm (Listing) wird einfach in den Speicher geladen und gestartet. Eine der integrierten Maschinenroutinen verschiebt das ganze Programm automatisch so im Speicher,

Register	Funktion
Multiregister 1	
Bit 0–2	: Bitweise vertikale Verschiebung des Bildschirms
Bit 3	: 24 oder 25 Zeilen
Bit 4	: Bildschirm abschalten
Multiregister 2	
Bit 0–2	: Bitweise horizontale Verschiebung des Bildschirms
Bit 3	: 38 oder 40 Spalten
Bit 4	: Multicolor-Flag
Bit 6	: PAL- oder NTSC-Norm
Bit 7	: Hardwaremäßige Invertierung ein/aus
Multiregister 3	
Bit 1	: Rasterinterrupt
Bit 2	: Lichtgriffelinterrupt
Bit 3	: Timer 1 interrupt
Bit 4	: Timer 2 interrupt
Bit 6	: Timer 3 interrupt
Bit 7	: Interrupt-Flag
Multiregister 4	
Bit 1	: Rasterinterrupt verhindern
Bit 2	: Lichtgriffelinterrupt verhindern
Bit 3	: Timer 1 abschalten
Bit 4	: Timer 2 abschalten
Bit 6	: Timer 3 abschalten
Multiregister 5	
Bit 0–3	: Lautstärke
Bit 4	: Tongenerator 1 einschalten
Bit 5	: Tongenerator 2 einschalten
Bit 6	: Tongenerator 2 Rauschen
Multiregister 6	
Bit 0–1	: Frequenz von Tongenerator 1 (Bit 8 und 9)
Bit 2	: Zeichensatzzugriff (ROM oder RAM)
Bit 3–7	: Adresse des Bitmapping-Speichers
Multiregister 7	
Bit 2–7	: Adresse des Zeichengenerators

Tabelle 1. Die Multiregister im TED

Reg. Nr.	Adresse	Funktion	Erläuterung
1	$FF00	Timer 1	(Low-Byte)
2	$FF01	Timer 1	(High-Byte)
3	$FF02	Timer 2	(Low-Byte)
4	$FF03	Timer 2	(High-Byte)
5	$FF04	Timer 3	(Low-Byte)
6	$FF05	Timer 3	(High-Byte)
7	$FF06	Multiregister 1	
8	$FF07	Multiregister 2	
9	$FF08	Tastatur	
10	$FF09	Multiregister 3	
11	$FF0A	Multiregister 4	
12	$FF0B	Rasterinterrupt	
13	$FF0C	Hardware Cursor Position	(Bit 8 und 9)
14	$FF0D	Hardware Cursor Position	(Bit 0 bis 7)
15	$FF0E	Tongenerator 1	
16	$FF0F	Tongenerator 2	(Bit 0-7)
17	$FF10	Tongenerator 2	(Bit 8 und 9)
18	$FF11	Multiregister 5	
19	$FF12	Multiregister 6	
20	$FF13	Multiregister 7	
21	$FF14	Adresse des Farb- und Videospeichers	
22	$FF15	Hintergrundfarbe	
23	$FF16	Vordergrund	
24	$FF17	Multicolor 1	
25	$FF18	Multicolor 2	
26	$FF19	Rahmenfarbe	
27	$FF1A	Bitmapping-Register	
28	$FF1B	Bitmapping-Register	
29	$FF1C	Abgetastete Rasterzeile	(Bit 8)
30	$FF1D	Abgetastete Rasterzeile	(Bit 0-7)
31	$FF1E	Rasterspalte	
32	$FF1F	Vertikal-Adresse	

Tabelle 2. Die Register des TED

daß der zu editierende Zeichensatz ab Adresse $1000 (wo sich eben normalerweise das Programm befindet) geladen werden kann.

Mit dem Cursor sucht man sich dann ein zu bearbeitendes Zeichen aus, wobei man sich wahlweise die Originalzeichen oder den geänderten Zeichensatz zeigen lassen kann. Durch einen Druck auf die RETURN-Taste wählt man das entsprechende Zeichen an. Dieses wird nun 8-fach vergrößert, damit man es – ebenfalls mit dem Cursor – editieren kann. Dabei sind die Funktionstasten das wichtigste Hilfsmittel: Mit F1 kann ein Punkt gesetzt werden (der Cursor bewegt sich dabei eine Position nach rechts), F2 löscht einen Matrixpunkt an der Zeigerposition. Mit der dritten Funktionstaste wählt man die Copy-Routine an. Damit läßt sich ein zweites Zeichen (wahlweise aus den beiden Original-Zeichensätzen oder aus dem zu editierenden) selektieren, das in das gerade bearbeitete Zeichen kopiert wird.

Übrigens kann man sich bei eventueller Fehlbedienung (egal in welchem Programmteil) mit der Escape(ESC)-Taste von dieser Funktion wieder befreien. Auch in das eigentliche Auswahlmenü (welches nach dem Programmstart erscheint) gelangt man durch Drücken der ESC-Taste.

Selbstverständlich ist auch das Laden und Speichern des Zeichensatzes möglich (diese Funktionen wählt man mit F6 und F7 an). Ferner ist ein Hilfsschirm, der über die wichtigsten Funktionen informiert, mittels der HELP-Taste abrufbar. Damit kann man sich noch über einige zusätzliche Funktionen des »Matrix-Editors« informieren, die daher hier nicht im einzelnen besprochen werden müssen.

Nun will man den gespeicherten Zeichensatz natürlich auch in seinen eigenen Programmen benutzen. Dabei ist zu beachten, daß man den Basic-Anfang (Adresse 44) vor dem Laden des Zeichensatzes verschieben muß, denn dort, wo normalerweise der Anfang des Basic-Speichers ist (bei $1000 beziehungsweise dezimal 4096), soll ja später der Zeichensatz abgelegt werden. Die Verschiebung wird mit »POKE 44,20 : POKE 20*256,0 : NEW« durchgeführt. Damit sind die Zeichendaten vor dem Zugriff des Basic-Interpreters geschützt und können von diesem nicht mehr überschrieben werden.

Wenn das Programm fertig erstellt ist, kann man Programm und Zeichensatz auch gleichzeitig abspeichern. Dazu muß zunächst ein kleiner Vorspann zu dem eigentlichen Programm hergestellt werden:

```
10 POKE 44,20
20 LOAD "TEIL 2" ,8,1 : REM " ,1,1" bei Datasette
```

Dieser wird nun zuerst gespeichert. Als nächstes wird der Basic-Anfang wie oben beschrieben verschoben. Dann wird der Zeichensatz mit »LOAD "Zeichensatz",8,1« von Disk oder Band (»,1,1«) nachgeladen. Schließlich und endlich lädt man dann das eigentliche Programm in den Basic-Speicher. Somit ist alles komplett. Diese Mischung von Zeichendaten und Programm wird nun hinter den zuvor generierten Vorspann gespeichert. Dazu gibt man ein:
»POKE 44,16 : SAVE "TEIL 2"« (»,8« für Diskette).

Damit kann man nun die mit dem »Matrix-Editor« erzeugten Zeichen in eigene Programme übernehmen.

Statt in hochauflösender Grafik mit Shapes zu arbeiten, kann man mit dieser Methode im normalen Textmodus bleiben und dennoch beliebige Grafiken auf den Bildschirm bringen, entweder mit PRINT-Anweisungen oder über POKEs in den Bildschirmspeicher. Das bringt nicht nur einen Geschwindigkeitsvorteil, sondern schont vor allem den begrenzten C16-Speicher. Mit über 11000 freien Bytes sollte sich schon einiges anfangen lassen.

(Christoph Sauer/ev)

```
10 PRINT"{CLR}";
20 CHAR1,6,2,"{RVSON,FLASHOFF}BITTE WART
EN !!!!!!"
30 POKE56,62:POKE55,223:CLR
40 TRAP 2570
50 PRINTCHR$(8)
60 ZA=4096:E$=CHR$(27)
70 FORT=1TO8:KEYT,CHR$(T+34):NEXT
80 IFPEEK(44)=20THEN140
90 FORT=16096TO16381
100 READD$:D=DEC(D$):S=S+D
110 POKET,D:NEXT
120 IFS<>35182THENPRINT"{CLR,FLASHON}PRU
EFSUMMENFEHLER !!{FLASHOFF}":END
130 SYS 16278:RUN
140 W=0:X1=0
150 POKE16174,109:POKE16271,64:SYS16128
160 COLOR 0,2,6:COLOR 4,5,3:PRINT"{CLR,P
URPLE,RVSON}";
170 FORT=1TO40:PRINT"{5SPACE}";:NEXT
180 CHAR 1,5,2,"{WHITE,2SPACE}M A T R I
X{4SPACE}E D I T O R{2SPACE}"
190 CHAR 1,10,4,"{BLACK}(C) 1985 BY C. S
AUER{RVOFF}"
200 SYS 16096
210 CHAR1,0,14,"DARSTELLUNG:"
220 CHAR1,0,16,"{RVSON}F1{RVOFF,3SPACE}:
 STANDARD{3SPACE}ZEICHENSATZ"
230 PRINT:PRINT"{RVSON,FLASHON}F2{FLASHO
FF,RVOFF,3SPACE}: UMDEFINIERTER ZEICHENS
ATZ"
240 PRINT"{RVSON}F3{RVOFF,3SPACE}: ZEICH
EN ZUR BEARBEITUNG LADEN"
```

Listing »Matrix-Editor« für den C16.
Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
250 CHAR1,0,21,"{RVSON}CLEAR{RVOFF}":PRI
NT:PRINT"{RVSON}HOME{SPACE,RVOFF}: ZEICH
ENSATZ INITIALISIEREN"
260 X=SX:Y=SY
270 GOSUB2670
280 IFA$="%"THENFA=1:SYS16122:GOTO1590
290 IFA$="{CLR}"THEN320
300 IFA$=CHR$(13)THENC=W:POKE2368+X1,113
:SX=X:SY=Y:GOTO370
310 GOTO270
320 COLOR 0,7,1:COLOR 1,7,1
330 CHAR1,0,21,"{WHITE}CLEAR"
340 CHAR1,0,22,"HOME :{SPACE,RVSON,FLASH
ON,5SPACE}SIND SIE SICHER ?{5SPACE,RVOFF
,FLASHOFF}"
350 GETKEYA$:IFA$="J"THENSYS16349
360 GOTO160
370 GOSUB2850:POKE65298,196
380 FORT=0TO7
390 BE(T)=PEEK(ZA+T+C*8)
400 BN=BE(T):GOSUB950
410 NEXT
420 COLOR 0,2,6:COLOR 4,4,5:PRINT"{CLR}"
;
430 PRINT"{WHITE,4RIGHT}∓∓∓∓∓∓∓∓∓∓"
440 PRINT"{4RIGHT}∓{GREEN}76543210{WHITE
}∓"
450 FORT=1TO8
460 PRINT""T"{LEFT}. ∓{BLACK}";BO$(T-1);
"{WHITE}∓ ";BE(T-1)
470 NEXT
480 PRINT"{4RIGHT}∓∓∓∓∓∓∓∓∓∓"
490 CHAR 1,22,6,"{RVSON,BLUE}STANDARD ZE
ICHEN"
500 CHAR 1,21,11,"{RVSON}SELBSTDEF. ZEIC
HEN"
510 POKE16271,106:POKE16174,123:SYS 1612
8:REM RASTERINTERRUPT
520 POKE2398,60:POKE3422,C
530 POKE2598,60:POKE3622,C
540 CHAR 1,21,11,"{RVSON}SELBSTDEF. ZEIC
HEN"
550 PRINTE$"M"
560 CHAR 1,0,17,"{LIG.BLUE}U************
****************************I"
570 PRINT"={4SPACE}F U N K T I O N S T A
 S T E N{5SPACE}="
580 PRINT"Q*******************W**********
**********W"
590 PRINT"={RVSON}F1{RVOFF}: PUNKT SETZE
N{3SPACE}={RVSON}F4{RVOFF}: DARSTELLUNG{
3SPACE}="
600 PRINT"={RVSON}F2{RVOFF}: PUNKT LOESC
HEN ={RVSON}F5{RVOFF}: ENDE{10SPACE}="
610 PRINT"={RVSON}F3{RVOFF}: KOPIEREN{7S
PACE}={RVSON}F6{RVOFF}: Z.SATZ{2SPACE}LA
DEN ="
620 PRINT"={RVSON}HE{RVOFF}: HILFSCHIRM{
5SPACE}={RVSON}F7{RVOFF}: ZS. SPEICHERN
="
630 PRINT"J******************************
**********K";
640 P=2:D=1
650 GETKEY A$
660 IFA$=CHR$(27)THEN150
670 IFA$="&"THEN1990
680 IFA$="'"THEN2110
690 IFA$="#"THENP=1:GOTO820
700 IFA$="$"THENP=0:GOTO820
710 IFA$="%"THEN1110
720 IFA$="{CLR}"THENFORT=0TO7:GOSUB2850:
POKEZA+C*8+T,0:NEXT:GOTO380
730 IFA$="{INST}"THENFORT=0TO7:GOSUB2850
:POKEZA+C*8+T,255:NEXT:GOTO380
740 IFA$="("THEN1580
750 IFA$=")"THEN1720
760 IFA$="*"THEN1370
770 FA=0
780 GOSUB2800
790 IFFA=0THEN820
800 IF A$="{HOME}"THENX=0:Y=0:GOTO820
810 GOTO650
820 IFD=1THEND=0:X=0:Y=0:AX=0:AY=0:Z=PEE
K(3157)
830 IFX>7THENX=0:Y=Y+40
840 IFX<0THENX=7:Y=Y-40
850 IFY>280THENY=0
860 IFY<0THENY=280
870 POKE3157+AX+AY,Z
880 Z=PEEK(3157+X+Y)
890 IFZ=81THENPOKE3157+X+Y,209:GOTO910
900 POKE3157+X+Y,190
910 AX=X:AY=Y
920 IFP=0THENGOSUB1020:X=X-1:Z=32:GOTO82
0
930 IFP=1THENGOSUB1020:X=X+1:Z=81:GOTO82
0
940 GOTO650
950 BO$(T)=""
960 FORB=1TO8
970 BN=BN/2
980 IF BN=INT(BN)THENBO$(T)=" "+BO$(T):E
LSEBO$(T)="@"+BO$(T)
990 BN=INT(BN)
1000 NEXT
1010 RETURN
1020 T=Y/40
1030 ON P+1GOTO1040,1050
1040 BE(T)=BE(T)AND255-2↑(7-X):GOTO1060
1050 BE(T)=BE(T)OR2↑(7-X)
1060 BN=BE(T):GOSUB950
1070 CHAR 1,0,T+2,"":P=2
1080 PRINT"{WHITE}"T+1"{LEFT}.{RIGHT}∓{B
LACK}"BO$(T)"{WHITE}∓{6SPACE,5LEFT}"BE(T
)
1090 POKEZA+C*8+T,BE(T)
1100 RETURN
1110 POKE16174,109:POKE16271,64:SYS16128
1120 COLOR 0,7,5:COLOR 4,13,2:PRINT"{CLR
,PURPLE,RVSON}";
1130 PRINT"{WHITE,RVSON,4SPACE}Z E I C H
 E N{5SPACE}K O P I E R E N{3SPACE,RVOFF
}"
1140 SYS16096
1150 CHAR1,0,14,"KOPIEREN VON:"
1160 CHAR1,0,16,"{RVSON}F1{RVOFF,3SPACE}
: STANDARD{3SPACE}ZEICHENSATZ"
1170 PRINT:PRINT"{RVSON,FLASHON}F2{FLASH
OFF,RVOFF,3SPACE}: UMDEFINIERTER ZEICHEN
SATZ"
1180 PRINT"{RVSON}F3{RVOFF,3SPACE}:"
1190 FL=2:ZS=0
1200 GOSUB2670
1210 IFFL=1THENCHAR1,0,18,"{RVSON}F3{RVO
FF,3SPACE}: UMSCHALTUNG GROSS/KLEIN"
1220 IFFL=2THENCHAR1,0,18,"{RVSON}F3{RVO
FF}":FORT=1TO35:PRINT" ";:NEXT
1230 IFA$=CHR$(13)THENPOKE2368+X1,113:GO
SUB2850:GOTO1280
1240 IFA$=CHR$(27)THEN420
1250 IFA$="%"ANDFL=1ANDZS=0THENPRINTCHR$
(14):ZS=4:GOTO1200:ELSE1200
1260 IFA$="%"ANDFL=1ANDZS=4THENPRINTCHR$
(142):ZS=0
1270 GOTO1200
1280 IFFL=1THENAD=208+ZS:GOTO1300
```

```
1290 AD=ZA/256
1300 W1=W*8:C1=C*8:W2=INT(W1/256):C2=INT
(C1/256)
1310 W1=W1-W2*256:POKE16115,AD+W2
1320 C1=C1-C2*256:POKE16118,ZA/256+C2
1330 POKE16114,W1:POKE16117,C1
1340 SYS16111:PRINTCHR$(142)
1350 GOTO380
1360 PRINT"{CLR,BLUE,SPACE,4RIGHT}ZEICHE
NSATZ:":FF=6
1370 SYS16122
1380 PRINT"{CLR,RVSON,5SPACE}H{2SPACE}I{
2SPACE}L{2SPACE}F{2SPACE}S{2SPACE}S{2SPA
CE}C{2SPACE}H{2SPACE}I{2SPACE}R{2SPACE}M
{4SPACE}"
1390 PRINT"{2DOWN,3SPACE}NM{CTRL-@}T{RIG
HT}X{WHITE,LIG.GREEN}"    THENFN
    EWAIT
1410 PRINT" NAUIAM{21SPACE}U***********I{
CTRL-@}Y{GREEN,F8,WHITE,LIG.GREEN}" RCLR
TAB(MID$GOTAB(END ODER   SWAIT  DWAIT  O
DER  PUDEF  "S$"  PUDEF
1430 PRINT"{3SPACE}MG{23SPACE}J**********
*K{CTRL-@}T{GREEN,SHFT-SPCE,WHITE,LIG.GR
EEN}"   RCLREND            XWAIT
1450 PRINT"{2DOWN,RVSON}ESC{RVOFF}: RUEC
KSPRUNG IN EINE HOEHERE EBENE"
1460 PRINT"{DOWN,RVSON}CLEAR{RVOFF,SPACE
,RVSON}SHIFT{RVOFF}: ZEICHEN(SATZ) LOESC
HEN"
1470 PRINT"{RVSON}HOME{SPACE,RVOFF,6SPAC
E}: CURSOR IN LINKE OBERE ECKE"
1480 PRINT"{DOWN,RVSON}INS{RVOFF}: ZEICH
EN FUELLEN"
1490 PRINT"{DOWN,RVSON}F3{RVOFF}: EINGE
WAEHLTES ZEICHEN WIRD KOPIERT"
1500 PRINT"{RVSON}F4{RVOFF}: SCHREIBMODU
S (AUSPROBIEREN DES ZS.)"
1510 PRINT"{RVSON}F5{RVOFF}: BEENDEN DES
 PROGRAMMS"
1520 PRINT"{RVSON}F6{RVOFF}: ZEICHENSATZ
 VON BAND ODER DISK LADEN"
1530 PRINT"{RVSON}F7{RVOFF}: ZEICHENSATZ
 ABSPEICHERN"
1540 PRINT"{DOWN,11SPACE,RVSON}ZURUECK M
IT{SPACE,FLASHON}ESC{FLASHOFF,RVOFF}";
1550 GETKEYA$
1560 IFA$<>CHR$(27)THEN1550
1570 GOTO420
1580 SYS16122
1590 PRINT"{CLR,RVSON,3SPACE}Z E I C H E
 N S A T Z{4SPACE}L A D E N{3SPACE}"
1600 PRINT"{3DOWN,RVSON,FLASHON}B{FLASHO
FF}AND ODER{SPACE,FLASHON}F{FLASHOFF}LOP
PY ?{RVOFF,2SPACE}";
1610 GETKEYA$
1620 IF A$="B"THENGN=1:PRINT"BAND":GOTO1
650
1630 IF A$="F"THENGN=8:PRINT"FLOPPY":GOT
O1650
1640 IF A$=CHR$(27)THEN420:ELSE1610
1650 GOSUB 1830
1660 PRINT"{2DOWN,RVSON,FLASHON}SIND SIE
 SICHER ?{FLASHOFF,RVOFF}"
1670 GETKEY A$:IFA$<>"J"THEN420
1680 LOAD F$,GN,1
1690 GOSUB1930
1700 IFFA=1THENFA=0:GOTO150
1710 GOTO420
1720 SYS16122
1730 PRINT"{CLR,RVSON}Z E I C H E N S A
T Z{2SPACE}S P E I C H E R N"
1740 PRINT"{3DOWN,RVSON,FLASHON}B{FLASHO
FF}AND ODER{SPACE,FLASHON}F{FLASHOFF}LOP
PY ?{RVOFF,2SPACE}";
1750 GETKEYA$
1760 IF A$="B"THENGN=1:PRINT"BAND":GOTO1
790
1770 IF A$="F"THENGN=8:PRINT"FLOPPY":GOT
O1790
1780 IF A$=CHR$(27)THEN420ELSE1750
1790 GOSUB1830
1800 POKE157,0:POKE158,20:POKE178,0:POKE
179,16:SYS61857
1810 GOSUB1930
1820 IFFF=1THENRETURN:ELSEGOTO420
1830 IFGN=1THENINPUT"{2DOWN,RVSON}FILENA
ME{RVOFF}";F$:GOTO1860
1840 INPUT"{2DOWN,RVSON}FILENAME{RVOFF }
(?=DIR) ";F$
1850 IFF$="?"THENPRINT"{CLR}":DIRECTORY:
GOTO1830
1860 IFLEN(A$)>16THENF$=LEFT$(F$,16)
1870 POKE174,GN:POKE171,LEN(F$)
1880 FORT=1TOLEN(F$)
1890 POKE1630+T,ASC(MID$(F$,T,1))
1900 NEXT
1910 POKE175,95:POKE176,6
1920 RETURN
1930 IF GN<>8THENRETURN
1940 IF DS=0THENRETURN
1950 PRINT"{2DOWN,FLASHON}FLOPPYFEHLER:{
FLASHOFF }"DS$
1960 PRINT"{DOWN,RVSON}WEITER MIT TASTE{
RVOFF}"
1970 GETKEY A$
1980 RETURN
1990 PRINT"{CLR,RVSON,3SPACE}D A R S T E
 L L U N G S M O D U S{4SPACE}"
2000 FOR T=4TO35:CHAR1,T,5,"*":NEXT
2010 FOR T=6TO19:CHAR1,4,T,"-{30SPACE}-"
:NEXT
2020 FOR T=4TO35:CHAR1,T,20,"*":NEXT
2030 CHAR1,4,5,"U":CHAR1,35,5,"I"
2040 CHAR1,4,20,"J":CHAR1,35,20,"K"
2050 CHAR1,10,23,"ENDE MIT{SPACE,RVSON}R
ETURN{RVOFF}"
2060 POKE2021,19:POKE2022,6:POKE2023,5:P
OKE2024,34
2070 SYS16122:POKE65298,PEEK(65298)AND25
1
2080 OPEN1,0:PRINT"{HOME}";
2090 INPUT#1,A$
2100 PRINT"{2HOME}":CLOSE1:GOTO420
2110 PRINT"{CLR,RVSON,12SPACE}E{3SPACE}N
{3SPACE}D{3SPACE}E{15SPACE}":FF=0
2120 PRINT"{3DOWN}WOLLEN SIE DEN ZEICHEN
S. ABSAVEN ? ";
2130 GETKEYA$
2140 IF A$=CHR$(27)THEN420
2150 IF A$="J"THENPRINT"JA":FF=1:GOSUB17
20
2160 IF A$<>"N"THEN2130
2170 POKE65298,PEEK(65298)OR4:SYS16122
2180 COLOR 0,7,5:COLOR 4,13,2
2190 PRINTE$"L{CLR,BLACK}"
2200 END
2210 DATA A2,7F,BA,9D,40,0D,A9,0A
2220 DATA 9D,40,09,CA,10,F4,60,A2
2230 DATA 07,BD,34,12,9D,00,10,CA
2240 DATA 10,F7,78,20,CE,F2,58,60
2250 DATA 78,A9,0D,8D,14,03,A9,3F
2260 DATA 8D,15,03,58,60,AD,09,FF
2270 DATA 29,02,F0,3D,AD,1C,FF,29
```

Listing «Matrix-Editor» für den C16 (Fortsetzung)

```
2280 DATA 01,D0,2C,AD,1D,FF,C9,91
2290 DATA B0,19,A9,C0,8D,12,FF,8D
2300 DATA 19,FF,AD,1D,FF,C9,93,90
2310 DATA F9,A9,C4,8D,12,FF,8D,19
2320 DATA FF,D0,0C,C9,B1,90,08,A9
2330 DATA C0,8D,12,FF,8D,19,FF,AD
2340 DATA 09,FF,29,02,F0,03,20,60
2350 DATA CE,2C,D8,07,10,0E,AD,01
2360 DATA FD,8D,D4,07,10,06,20,95
2370 DATA EA,20,5B,EA,20,E4,E3,AD
2380 DATA 09,FF,29,02,F0,25,8D,09
2390 DATA FF,2C,0B,FF,A9,C3,50,18
2400 DATA 20,BF,CF,20,CD,CE,A5,FB
2410 DATA 48,A9,00,85,FB,08,58,20
2420 DATA 11,DB,28,68,85,FA,A9,62
2430 DATA 8D,0B,FF,4C,BE,FC,A6,2D
2440 DATA 86,5A,A4,2E,84,5B,20,D0
2450 DATA 3F,86,58,86,2D,84,59,84
2460 DATA 2E,A6,2B,86,5F,A4,2C,84
2470 DATA 60,20,D0,3F,86,2B,86,A6
2480 DATA 84,2C,84,A7,20,C0,88,20
2490 DATA 18,88,A5,A6,D0,02,C6,A7
2500 DATA C6,A6,A9,00,A8,91,A6,60
2510 DATA 84,A8,18,8A,69,00,AA,A5
2520 DATA A8,69,04,A8,60,A9,00,85
2530 DATA A6,85,A8,A9,D0,85,A7,A9
2540 DATA 10,85,A9,A2,04,A0,00,B1
2550 DATA A6,91,A8,C8,D0,F9,E6,A7
2560 DATA E6,A9,CA,D0,F0,60
2570 IF ER=4THENRESUMENEXT
2580 IF ER=35THEN2110
2590 IF ER=30THEN2110
2600 IF ER=5THENPRINT"{2DOWN,FLASHON}FLO
PPY EINSCHALTEN !!!":FORT=1TO1000:NEXT:R
ESUME1700
2610 PRINTERR$(ER)" IN "EL
2620 GOTO 2110
2630 X=SX:Y=SY
2640 POKE2368+X1,10
2650 POKE2368+W,241:X1=W
2660 RETURN
2670 POKE2368+X1,10
2680 POKE2368+W,241:X1=W
2690 GETKEYA$
2700 GOSUB2800
2710 IFX>40THENX=0:Y=Y+40
2720 IFX<0THENX=40:Y=Y-40
2730 IFY>120THENY=120
2740 IFY<0THENY=0
2750 W=Y+X
2760 IF W>127THENW=127
2770 IFA$="#"THENSYS16122:CHAR1,0,16,"{R
VSON,FLASHON}F1{FLASHOFF,RVOFF}":PRINT:P
RINT"{RVSON}F2{RVOFF}":FL=1
2780 IFA$="$"THENSYS16128:CHAR1,0,16,"{R
VSON}F1{RVOFF}":PRINT:PRINT"{RVSON,FLASH
ON}F2{FLASHOFF,RVOFF}":FL=2
2790 RETURN
2800 IFA$="{DOWN}"ORA$="X"THENY=Y+40:RET
URN
2810 IFA$="{RIGHT}"ORA$="D"THENX=X+1:RET
URN
2820 IFA$="{UP}"ORA$="E"THENY=Y-40:RETUR
N
2830 IFA$="{LEFT}"ORA$="S"THENX=X-1:RETU
RN
2840 FA=0:RETURN
2850 CHAR1,32,4,"{RVSON,FLASHON}WORKING{
RVOFF,FLASHOFF}"
2860 RETURN
```

© 64'er

Listing »Matrix-Editor« für den C16 (Schluß)

Grafikbeispiel für den C 16

Unsere »Blockgrafik« ist ein Beispiel dafür, daß Sie auch mit dem vorhandenen Zeichensatz fantastische Bilder erzeugen können.

Es muß nicht immer HiRes sein. Gerade beim C 16 ist die Verwendung der hochauflösenden Grafik (HiRes) kritisch, weil ein erheblicher Speicherbereich für die Programmierung verloren geht. Diese Blockgrafik (Bild) zeigt, wie es auch anders geht. Das untenstehende Listing ist das dazugehörige Basic-Programm. (Jörg Gerjets/kn)

```
10 REM **********************************
15 REM **                              **
20 REM ** 'BLOCKGRAPHIK'               **
25 REM **                              **
30 REM **********************************
35 REM **                              **
40 REM **    FUER  C 16/116 & C 64     **
45 REM **                              **
50 REM **********************************
55 REM ** JOERG GERJETS / 2980 NORDEN **
60 REM **                              **
65 REM **      (C) BY DPS MCMLXXXV     **
70 REM **********************************
75 REM ** BEI C 16/116 DURCH DRUECKEN **
80 REM **    DER TASTEN '@','+','-'    **
85 REM **         & '*' BEWEGUNG       **
90 REM **          [ESC] = ENDE        **
95 REM **********************************
100 PRINT"{2HOME}":PRINTCHR$(142):COLOR4,1:COLOR0,
2
110 PRINT"{CLR}";
120 PRINT"{BLACK,RVSON,4SPACE,BLUE}F 2 3 4{BLACK,2
SPACE}#########{4SPACE}↑ ↑↑ ↑{SPACE,BLUE,RVSON }2
4{SPACE,BLACK,2SPACE}"
130 PRINT"{RVSON,4SPACE,BLUE}U{2SPACE,BROWN,3SPACE
,BLUE,SPACE,RVOFF,CYAN}UCCCI{BLACK}UCCCCCI{LIG.BLU
E}Z{BLACK}W{RVSON}URRRRRRI{BLUE,RVSON}0{BROWN }B{S
PACE,BLUE}6{BLACK,2SPACE}"
140 PRINT"{RVSON,4SPACE,BLUE}E 1{BROWN }U{SPACE,BL
UE}5{RVOFF,CYAN}B{RED}Q{SPACE,GREEN}Q{CYAN}B{BLACK
}B{RVSON,ORANGE }M-3{SPACE,RVOFF,BLACK}B{BROWN}UI{
BLACK,RVSON}O+++++++P{BLUE,RVSON,SPACE,BROWN }H{SPA
CE,BLUE,SPACE,BLACK,2SPACE}"
150 PRINT"{RVSON,4SPACE,BLUE}L{2SPACE,BROWN,2SPACE
}M{BLUE,SPACE,RVOFF,CYAN}B{SPACE,GREEN}Q{YELLOW}Q{
CYAN}B{BLACK}JCCCCCK{BROWN}BB{BLACK,RVSON}O+++++++P
{BLUE,RVSON}HT{BROWN,2SPACE,BLUE}8{BLACK,2SPACE}"
160 PRINT"{RVSON,4SPACE,BLUE,2SPACE}0 7 6{RVOFF,CY
AN}JCCCK{RVSON,RED}K30.37{RVOFF}K{BROWN}JK{BLACK,R
VSON}JEEEEEEK{BLUE,RVSON,2SPACE}10{SPACE,BLACK,2SP
ACE}"
170 PRINT"£{ORANGE}D{RVSON}YYYYYYYYYYYYYYYYYYYYYYY
YYYYYYYYYYYYY{RVOFF}F{BLACK}W"
180 PRINT"{ORANGE,SPACE,RVSON,SPACE,RVOFF,17SPACE,
RVSON,2SPACE,RVOFF,4SPACE,BLACK,11SPACE}.{SPACE,OR
ANGE,RVSON }"
190 PRINT"N{RVSON,SPACE,RVOFF,5SPACE,BLACK,RVSON}W
{RVOFF,ORANGE,11SPACE,RVSON,2SPACE,RVOFF,2SPACE,BL
ACK,5SPACE},{9SPACE,ORANGE,RVSON,SPACE,RVOFF}H"
200 PRINT"T{RVSON,SPACE,RVOFF,4SPACE,BLACK,SPACE,R
VSON,2SPACE}YYT W{RVOFF,ORANGE,5SPACE,RVSON,2SPACE
,RVOFF,13SPACE,ORANGE}X{3SPACE,ORANGE,RVSON,SPACE,
RVOFF}J"
210 PRINT"{RVSON}K{SPACE,RVOFF,6SPACE,BLACK}Y{RVSO
N}F{RVOFF}U{RVSON}F{RVOFF}UU{ORANGE,5SPACE,RVSON,2
SPACE,RVOFF,13SPACE,RED,RVSON}F{RVOFF,3SPACE,ORANG
E,RVSON,SPACE,RVOFF}K"
220 PRINT"{RVSON}K{SPACE,RVOFF,17SPACE,RVSON,2SPAC
E,RVOFF,4SPACE,BLACK}X{8SPACE,RED,RVSON}F{RVOFF,3S
PACE,ORANGE,RVSON,SPACE,RVOFF}K"
230 PRINT"T{RVSON,SPACE,RVOFF,13SPACE,BLACK}O{ORAN
GE,3SPACE,RVSON,2SPACE,RVOFF,3SPACE,BLACK,RVSON}Y{
SPACE,RVOFF,8SPACE,RED,RVSON}F{RVOFF,3SPACE,ORANGE
,RVSON,SPACE,RVOFF}J"
```

Listing. Basic-Programm zur Blockgrafik.
Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
240 PRINT"W{RVSON,SPACE,RVOFF,BROWN}DPDPD{2SPACE,B
LUE }D D{2SPACE,BLACK}T{RVSON,3SPACE,ORANGE,2SPACE
,BLACK,6SPACE,RVOFF}£{BLUE }D{2SPACE,BROWN}DT{RVSO
N }U{RVOFF}F{SPACE,ORANGE,RVSON,SPACE,RVOFF}H"
250 PRINT"{SPACE,RVSON,SPACE,BLUE,17SPACE,ORANGE,2
SPACE,BLUE,17SPACE,ORANGE }"
260 PRINT" C{RVSON}PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP
PPPPP{RVOFF}V"
270 PRINT" FFF{SPACE,GREEN}F{2SPACE,RED}UCCCCI{SPA
CE,BLACK}UCCCCCCCCCFCCI -{BLUE}RADAR{BLACK}- UCI"
280 PRINT" H0H{GREEN}Q{RVSON}N{RVOFF}L{SPACE,RED}B
{RVSON}W{2SPACE}1{RVOFF}B{GREEN}<{BLACK}B{LIG.GREE
N,RVSON}£ D{2SPACE}D T{RVOFF,BLACK}B{RED}QQ{BLACK}
B{SPACE,RED,RVSON}£{2SPACE}B{2SPACE}T{RVOFF,BLACK
}D W"
290 PRINT"{LIG.BLUE }>{SPACE,RVSON,GREEN}£{3SPACE}
T{RVOFF,RED}JCCFCFI{BLACK}B{LIG.GREEN,RVSON }DD{2S
PACE}DD{SPACE,RVOFF,BLACK}B{SPACE,ORANGE}↑{BLACK}B
{SPACE,RED,RVSON,5SPACE}H{SPACE,RVOFF,BLACK }D W"
300 PRINT"{SPACE,GREEN}F{RVSON}£{5SPACE}T{RVOFF}F{
SPACE,RED}B{LIG.BLUE,RVSON,2SPACE,RVOFF,RED}B{BLAC
K}B{LIG.GREEN,RVSON,2SPACE}D{2SPACE}D{2SPACE,RVOFF
,BLACK}B{GREEN}Q{SPACE,BLACK}BU{RED,RVSON,4SPACE}N
{2SPACE,RVOFF,BLACK}ID{YELLOW}D{BLACK}W"
310 PRINT"{GREEN}W{RVSON}W ←CHC{2SPACE}E{RVOFF}H{R
ED}B{LIG.BLUE,RVSON,2SPACE,RVOFF,RED}B{BLACK}B{LIG
.GREEN,RVSON}F FQLF F{RVOFF,BLACK}B{BLUE}Q{SPACE,B
LACK}BB{RED,RVSON}C{2SPACE}H{2SPACE}C{RVOFF,BLACK}
BD{YELLOW,RVSON,SPACE,RVOFF,BLACK}W"
320 PRINT"{SPACE,GREEN}YT{RVSON,5SPACE,RVOFF}£Y{SP
ACE,RED}JCCK{BLACK}B{RVSON}Y D{2SPACE}D Y{RVOFF}B{
SPACE,YELLOW}Q{BLACK}BJ{RED,RVSON,7SPACE,RVOFF,BLA
CK}ED{YELLOW,RVSON,SPACE,RVOFF,BLACK}W"
330 PRINT"{SPACE,ORANGE}F{SPACE,GREEN}T{RVSON,3SPA
CE,RVOFF}£{ORANGE}←H{SPACE,PURPLE}DPS{ORANGE}-{BLA
CK}B{RVSON }DD{2SPACE}DD{SPACE,RVOFF}B{ORANGE}H{CY
AN}+{BLACK}B{SPACE,RED,RVSON,7SPACE,RVOFF,BLACK }D
{YELLOW,RVSON,SPACE,RVOFF,BLACK}W"
340 PRINT"{ORANGE}W{PURPLE}Q{ORANGE}H{SPACE,GREEN}
P{RVSON}S{RVOFF}Q{RED}↑CCCCCCS{BLACK}B{RVSON,2SPAC
E}D{2SPACE}D{2SPACE,RVOFF}B{SPACE,PURPLE}Q{BLACK}B
{SPACE,RED}T{RVSON,2SPACE}B{2SPACE,RVOFF}£{BLACK }
D{YELLOW,RVSON,SPACE,RVOFF,BLACK}W"
350 PRINT"{BROWN}W{RVSON}#4{RVOFF}H{SPACE,GREEN}Y{
BLACK,SPACE,RED}B{BLACK}>>↑{BLUE}X{BLACK}<<{RED}B{
BLACK}BT{RVSON,6SPACE,RVOFF}£B{RED}*{BLACK}*B{3SPA
CE}JCK{3SPACE}JCK{4SPACE}"
360 PRINT"{RVSON}T{RVOFF }DDDD{SPACE,RED}7CCCCCCY{
BLACK}JCCCCCCCCFCCK{SPACE,LIG.GREEN}H{LIG.BLUE}EEE
EE{LIG.GREEN}H{BLACK }← ";:POKE4071,233
370 REM
```

```
380 REM                                    ©64'er
390 REM
400 REM *********************************
410 REM ** BEWEGUNG - NUR C 16/116 ! **
420 REM *********************************
430 DO:GETKEYA$
440 IFA$="*"THENCHAR1,34,08,"{FLASHON,ORANGE}X{BLA
CK}":CHAR1,37,24,"{FLASHON,BLACK}←":CHAR1,26,19,"{
FLASHON,RED}Z"
450 IFA$="+"THENCHAR1,13,3,"{FLASHON,GREEN}Q":CHAR
1,1,22,"{FLASHON,PURPLE}Q"
460 IFA$="-"THENCHAR1,25,23,"{FLASHON,RED}*":CHAR1
,23,1,"{FLASHON,PURPLE}Z"
470 IFA$="@"THENPRINT"{HOME,2RIGHT,6DOWN}"CHR$(27)
"T"
480 IFA$="@"THENPRINT"{HOME,16RIGHT,4DOWN}"CHR$(27
)"B";:GOTO510
490 IFA$=CHR$(27)THEN600
500 LOOP
510 FORI=1TO4:FORJ=1TO1000:NEXT:PRINT:NEXT:LOOP
600 PRINT"{2HOME,CLR,BLACK,4DOWN,4SPACE}**{2SPACE}
(C) BY DON'T PANIC SOFT{2SPACE}**"
700 REM *********************************
710 REM *********************************
720 REM **   ACHTUNG : AENDERUNGEN      **
730 REM **        FUER C 64             **
740 REM *********************************
750 REM ** POKE 4071,233 IN ZEILE 360 **
760 REM **  MUSS DURCH POKE 2023,233    **
770 REM **        ERSETZT WERDEN        **
780 REM *********************************
790 REM **   COLOR 0,2 IN ZEILE 100     **
800 REM **  MUSS DURCH POKE 53281,1     **
810 REM **        ERSETZT WERDEN        **
820 REM *********************************
830 REM **   COLOR 4,1 IN ZEILE 100     **
840 REM **  MUSS DURCH POKE 53280,0     **
850 REM **        ERSETZT WERDEN        **
860 REM *********************************
870 REM **   DAS STEUERZEICHEN FUER     **
880 REM **    HELLBLAU MUSS DURCH       **
890 REM **    NORMAL-BLAU ERSETZT       **
900 REM **    NUR WICHTIG IN ZEILE      **
910 REM **         280 - 310            **
920 REM *********************************
930 REM **  ZEILE 430 MUSS LAUTEN :     **
940 REM **  430 GETA$:IFA$=""THEN430    **
950 REM **  ZEILE 440 - 510 DUERFEN     **
960 REM **  NICHT UEBERNOMMEN WERDEN    **
970 REM *********************************
980 REM *********************************
```

Listing. Basic-Programm zur Blockgrafik. Bitte beachten Sie die Eingabehinweise auf Seite 76.

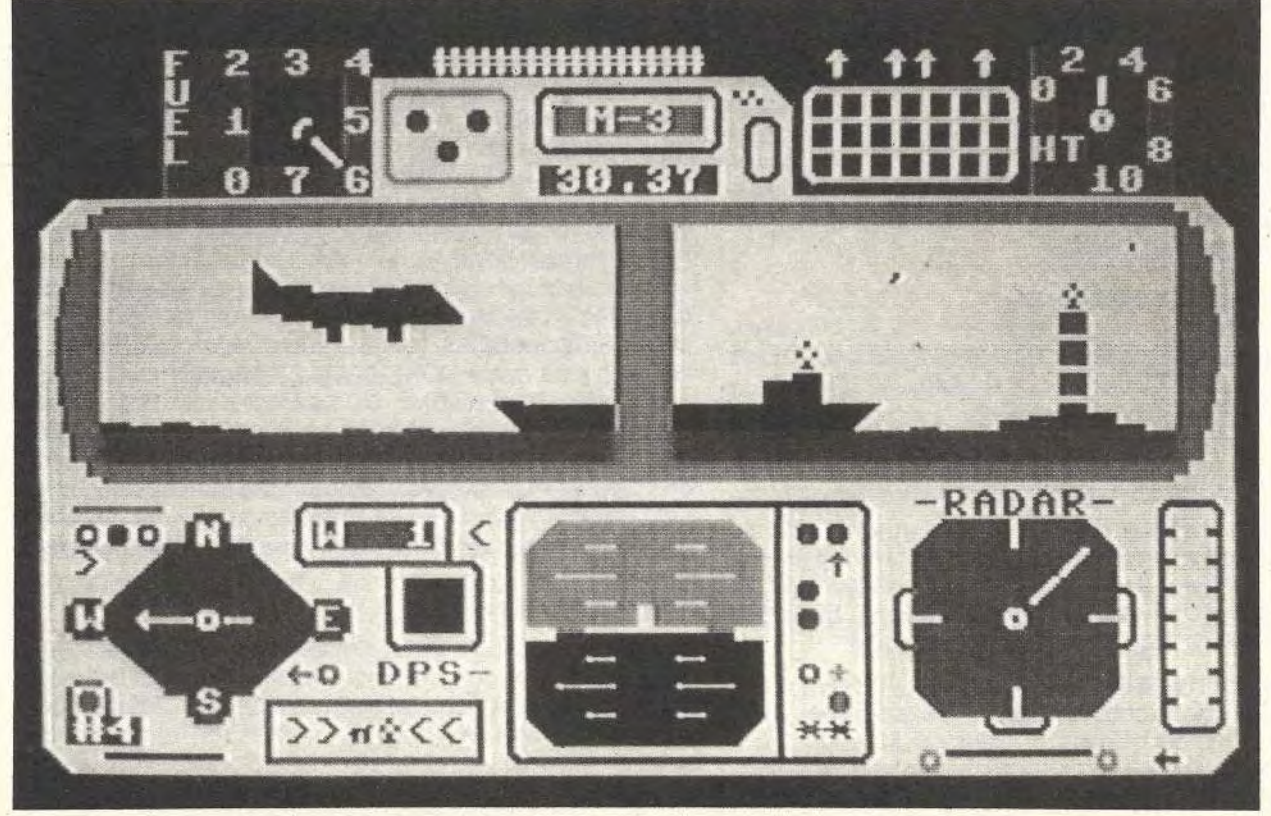


Bild. Blockgrafik für den C16

Hyper-Graphics

Das Programm »Hyper-Graphics« erlaubt über SYS-Befehle vom Basic aus Grafiken mit einer Auflösung von 208 * 256 = 53248 Bildpunkten zu erstellen. Damit stößt der VC 20 auf seine alten Tage noch in Bereiche des ZX-Spectrum oder Apple II vor.

Immer wenn über die VC-20-Grafik geschrieben wurde, bekam man bisher vorgerechnet, daß die Auflösung nicht 28160 Punkte überschreiten kann. Um es gleich vorwegzunehmen: In der Tat ist die echte Maximalauflösung mit 160 * 192 = 30720, wie sie durch entsprechendes Umgestalten der Data-Becker-Grafik aus dem Buch »Tips & Tricks für den VC 20« gelang, weit von jenen 53248 Punkten entfernt, doch läßt sich diese Auflösung durch einen Trick erreichen. Doch zuerst sei die Frage beantwortet, warum es überhaupt erstrebenswert ist, eine solche Auflösung mit so »krummen« Werten zu entwerfen: Die 256 Punkte in vertikaler Richtung werden benötigt, um komplette 8-Bit-Werte darstellen zu können. Die 208 Punkte in horizontaler Richtung ergeben sich aus dem Umstand, daß nicht mehr auf den normalen Fernsehbildschirm paßt. Bei manchen älteren Geräten kann es vorkommen, daß die Bildschirmecken der Grafik abgeschnitten werden oder daß am Bildschirmrand Verzerrungen auftreten. Der Vorteil vertikal 256 Punkte darstellen zu können, wird besonders dadurch wertvoll, daß die meisten mit dem Computer erfaßten Meßwerte 8-Bit-Werte zum Beispiel von Analog/Digital-Wandlern sind. So habe ich mir beispielsweise einen solchen A/D-Wandler mit dem Baustein ZN 427 gebaut und damit in Abhängigkeit von der Zeit die Spannung an einem sich entladenden Kondensator gemessen. Da »Hyper-Graphics« vertikal 256 Punkte setzen kann, brauchten die Daten nicht gerundet werden, was einen gleichmäßigeren und genaueren Graphen ergab, als dies mit Computern, wie zum Beispiel dem C64, dem C128 oder dem Schneider-Computer, möglich ist, da diese zwar der Punktezahl nach eine noch wesentlich bessere Grafik bieten, aber in der Vertikalen nur 200 Punkte darstellen können, was einen deutlichen Genauigkeitsverlust bedeutet, besonders wenn man versucht, 256 Werte auf die 200 möglichen Punkte umzurechnen, da hier bedingt durch das erforderliche Runden die gemessenen Daten arg verzerrt und verfremdet wiedergegeben werden.

Das »Hyper-Graphics«-Verfahren

Das Verfahren ist im Prinzip so alt wie der VC20 selbst, denn der normale Bildschirm arbeitet ähnlich: Er besteht aus 22 * 23 Zeichen, was bei einer 8*8-Matrix 32384 Punkte ergibt. Dies ist nur durch Mehrfachbenutzung der einzelnen Buchstaben und Zeichen möglich. Da bei den meisten Darstellungen nur einige Graphen gleichzeitig dargestellt werden müssen, kann man in die übrigen Bildschirmstellen ein und dasselbe punktfreie Zeichen schreiben und erhält so eine Grafik, die einer erheblich höher aufgelösten entspricht, obwohl der Computer genaugenommen nur etwas mehr als den halben Bildschirm »mit Punkten füllen« kann. Es kommt eben nur auf die richtige Verteilung der Punkte an!

Anwendung von Hyper-Graphics

Als erstes ist das Programm abzutippen, und zwar wahlweise als Basic-Lader (Listing 1) oder als Maschinenprogramm (Listing 2). Der Basic-Lader bietet den Vorteil, die eingePOKEten Daten durch eine Checksumme zu überprüfen. Trotzdem kann es aber zu unerkannten Fehlern kommen; deshalb sollte das Programm unbedingt vorher gespeichert werden, zumal der Lader durch Starten des Maschinenprogramms auf jeden Fall zerstört wird. Ist das Programm einmal korrekt eingegeben und erfolgreich gestartet, meldet sich der VC 20 mit einer Bildschirmzeile mehr, die den Benutzer in roter, reverser Schrift daran erinnert, daß er »Hyper-Graphics« eingeschaltet hat. Diese Zeile läßt sich mit dem normalen Editor nicht ändern und bleibt auch nach STOP/RESTORE erhalten. Wem die Farbe aber nicht gefällt, der kann in den Basic-Lader vor den SYS-Befehl ein POKE 9487, Punktfarbe setzen und damit den Schriftzug seinen Vorstellungen anpassen.

Folgende Befehle stellt das Programm zur Verfügung (siehe auch Listing 3):

SYS IN: Schaltet Grafik-Modus ein

SYS CL: Löscht den Grafik-Bildschirm

SYS OF: Schaltet den Grafik-Modus aus

SYS SE, X, Y, Punktfarbe: Setzt einen Punkt

SYS RE, X, Y: Löscht einen Punkt

SYS GS, »Filename«, Devicenumber: Speichert Grafik

SYS GL, »Filename«, Devicenumber: Lädt Grafik

Die Variablen haben folgende Werte beziehungsweise Bedeutungen:

IN = 9330: CL = IN+3: SE = CL+3: RE = SE+3: GS = RE+3: GL = GS+3: OF = GL+3:

0 <= X <= 207 und bedeutet die X-Koordinate
0 <= Y <= 255 und bedeutet die Y-Koordinate

Die Punktfarbe entspricht der sonst üblichen Buchstabenfarbe, »Devicenumber« gibt das Ein-/Ausgabegerät an.

Da der Grafik nur maximal 255 - 26 = 229 Zeichen zur Verfügung stehen (der Videochip läßt nicht mehr als 256 Zeichen zu, die Zeichen 0 bis 25 überschneiden sich mit dem Video-RAM, und Zeichen 26 ist das Grafik-Leerzeichen), kann es vorkommen, daß die Grafikkapazität vor Fertigstellen der Grafik erschöpft ist; der Computer meldet dann einen »TOO MANY CHARACTERS ERROR«, was dem Benutzer signalisieren soll, daß zur Darstellung der Grafik mehr als die vorhandenen 229 Zeichen benötigt werden (wie das Demo-Programm aber beweist, muß man den Bildschirm schon sehr vollschreiben, um diese Fehlermeldung zu erreichen). Soll diese Fehlermeldung ignoriert werden, um wenigstens den bisher fertiggestellten Teil der Grafik weiter anzuzeigen, so läßt sich dies durch POKE 9609,27 erreichen, indem man nach jedem Punktsetzen die Speicherstelle 680 ausliest: Solange ihr Wert größer als 26 ist, können beruhigt weitere Punkte gesetzt werden, ist ihr Wert aber 26, so können nur dort weitere Punkte gesetzt werden, wo bereits Zeichen auf dem Bildschirm sind. Um festzustellen, ob an der betreffenden Stelle bereits ein Zeichen im Video RAM steht, gibt man

```
SYS 9663,X,Y : SYS 9707 : A = PEEK (256*PEEK(252)+
PEEK(251))
```

ein. Ist A=26, dann kann an die betreffende Stelle kein Punkt mehr gesetzt werden. Die Punktlöschroutine prüft bei jedem Punktlöschen, ob das aktuell bearbeitete Zeichen leer geworden ist und so durch das Leerzeichen ersetzt werden kann. Ist dies der Fall, dann wird das freigewordene Zeichen

Fortsetzung Text auf Seite 62

Listing 1. DATA-Lader zu »Hyper-Graphics«

```
10 REM ******************************                <064>
20 REM *       HYPER-GRAPHICS       *                <200>
30 REM * (C) 1985 BY JOACHIM BROCKE *                <026>
40 REM *  DIETRICH-BONHOEFFERSTR.5  *                <087>
50 REM *  4790 PADERBORN            *                <104>
60 REM ******************************                <114>
70 PRINT"{CLR,RVSON}>>> HYPER-GRAPHICS <<<"          <168>
80 PRINT TAB(7)"(C) 1985"SPC(31)"BY JOACHIM BROCKE"  <118>
90 PRINT"{DOWN,RVSON,4SPACE}BITTE WARTEN !{4SPACE}"  <080>
100 REM DATEN EINPOKEN                               <065>
110 FOR I=0 TO 831                                   <201>
120 READ X:A=A+X                                     <201>
130 POKE I+8192+1024,X                               <111>
140 NEXT                                             <150>
150 REM TEST AUF KORREKTE DATENEINGABE               <201>
160 IF A<>97379 THEN PRINT"{RVSON}FEHLER IN DATAZEILEN !":END   <142>
170 REM PROGRAMMSTART                                <071>
180 SYS 9980                                         <157>
190 REM MASCHINENPROGRAMM IN DATA-ZEILEN             <102>
200 DATA 190,190,190,160,136,153,144,133,146,173,135,146,129,144,136,137,131   <142>
210 DATA 147,160,188,188,188,63,84,79,79,32,77,65,78,89,32,67,72,65,82,65,67   <180>
220 DATA 84,69,82,83,0,26,52,78,104,130,156,182,208,234,4,30,56,82,108,134,16  <086>
230 DATA 16,16,16,16,16,16,16,16,16,17,17,17,17,17,17,128,64,32,16,8,4,2,1,9   <130>
240 DATA 19,26,33,255,204,40,67,41,32,49,57,56,53,32,66,89,32,74,79,65,67,72   <172>
250 DATA 73,77,32,66,82,79,67,75,69,76,135,36,76,53,37,76,112,37,76,27,38,76   <068>
260 DATA 129,38,76,162,38,76,220,36,173,5,144,201,204,240,51,160,6,185,81,36   <167>
270 DATA 153,255,143,136,208,247,162,208,189,255,31,157,255,15,189,207,32,157  <080>
280 DATA 207,16,189,159,33,157,255,147,189,111,34,157,207,148,202,208,229,162  <239>
290 DATA 112,189,63,35,157,159,17,202,208,247,96,40,[illegible],41,32,49,57,56,53,32,2   <210>
300 DATA 25,32,10,15,1,3,8,9,13,32,2,18,15,3,11,5,173,5,144,201,192,240,43,162 <089>
310 DATA 208,189,255,15,157,255,31,189,207,16,157,207,32,189,255,147,157,159   <167>
320 DATA 33,189,207,148,157,111,34,202,208,229,162,112,189,159,17,157,63,35,202  <196>
330 DATA 208,247,32,95,229,160,2,162,22,189,255,35,157,249,17,152,157,249,149  <017>
340 DATA 202,208,243,160,6,185,227,237,153,255,143,136,208,247,169,34,141,1,144  <113>
350 DATA 169,48,141,3,144,96,169,26,162,208,157,255,15,157,207,16,157,255,31   <233>
360 DATA 157,207,32,202,208,241,162,96,169,0,157,159,17,157,159,33,202,208,247  <114>
370 DATA 169,18,133,252,169,0,133,251,162,14,168,145,251,136,208,251,230,252   <241>
380 DATA 202,208,246,169,255,141,168,2,96,32,191,37,32,253,206,32,158,215,142  <095>
390 DATA 173,2,32,235,37,201,26,208,12,173,168,2,201,27,144,32,206,168,2,145   <205>
400 DATA 251,32,254,37,17,253,145,253,165,252,24,105,132,133,252,172,169,2,173  <181>
410 DATA 173,2,145,251,96,169,0,133,62,32,220,36,169,22,160,36,32,30,203,76,98  <030>
420 DATA 196,32,220,36,162,14,76,55,196,32,253,206,32,158,215,224,208,176,238  <249>
430 DATA 138,74,74,74,141,169,2,138,41,7,141,170,2,32,253,206,32,158,215,138   <252>
440 DATA 74,74,74,74,141,171,2,138,41,15,141,172,2,96,172,171,2,185,42,36,133  <112>
450 DATA 251,185,58,36,133,252,172,169,2,177,251,96,168,74,74,74,74,24,105,16  <053>
460 DATA 133,254,152,10,10,10,10,24,133,253,173,170,2,170,189,74,36,172,172,2  <135>
470 DATA 96,32,191,37,32,235,37,201,26,240,94,141,174,2,32,254,37,73,255,49,253  <056>
480 DATA 145,253,160,0,177,253,208,73,200,192,16,208,247,172,169,2,169,26,145  <013>
490 DATA 251,238,168,2,162,208,189,207,16,205,168,2,240,17,189,255,15,240,3,202  <086>
500 DATA 208,240,173,174,2,157,255,15,76,103,38,173,174,2,157,207,16,165,253   <223>
510 DATA 133,251,165,254,133,252,173,168,2,32,254,37,160,0,177,253,145,251,200  <040>
520 DATA 192,16,208,247,96,32,220,36,32,253,206,32,209,225,173,168,2,141,255   <201>
530 DATA 35,162,0,160,36,138,133,253,169,18,133,254,169,253,133,185,76,216,255  <233>
540 DATA 32,220,36,32,253,206,32,209,225,169,1,133,185,169,0,32,213,255,173,255  <148>
550 DATA 35,141,168,2,96,72,138,72,152,72,173,29,145,16,37,45,30,145,170,41    <181>
560 DATA 2,240,32,44,17,145,32,52,247,32,225,255,208,18,32,82,253,32,240,38,32  <253>
570 DATA 249,253,32,24,229,32,220,36,108,2,192,76,86,255,76,222,254,169,187,141  <175>
580 DATA 24,3,169,38,141,25,3,96,120,32,240,38,173,0,3,141,58,39,173,1,3,141   <095>
590 DATA 59,39,169,45,141,0,3,169,39,141,1,3,32,220,36,169,39,133,44,141,130   <074>
600 DATA 2,169,64,133,43,141,129,2,32,68,198,88,76,116,196,142,57,39,32,220,36  <030>
610 DATA 174,57,39,108,58,39,28,58,196,0,0,0,0                                 <066>
```

Fortsetzung von Seite 60

gegen das »größte« nicht freie Zeichen ausgetauscht und dieses Zeichen wieder freigegeben, indem der Zähler für die freien Zeichen erhöht wird.

Zum Demo-Programm (Listing 4):
Es enthält folgende 4 Teile:

1. Eine 3D-Effekt-Grafik; die Formel hierzu ist dem oben erwähnten Data-Becker-Buch entnommen und wurde für »Hyper-Graphics« umgeschrieben.
2. Die für Grafikprogramme obligatorische Sinuskurve wird dargestellt.
3. Eine aus Rechtecken bestehende Zentralperspektive wird dargestellt.
4. Es wird mit hochauflösender Grafik in doppelthoher Schrift ein Text geschrieben und dann perspektivisch ein Oktaeder in Durchsichtszeichnung gezeichnet (dieser Programmteil ist Bestandteil eines Programmes, das auf einer Ausstellung unseres Chemie-Leistungskurses lief, und stieß im großen und ganzen auf positive Kritik).

Die einzelnen Programmteile werden durch Druck der angezeigten Zahlentaste gewählt, danach sind Rahmen-, Hintergrund- und Punktfarbe im bekannten Rahmen der VC-20-Möglichkeiten einzugeben, wobei auf korrekte Eingabe getestet wird. Da das Demo-Programm nur zum Ansehen dient, wurde, um die Abtipparbeit zu erleichtern, weitgehend auf REMs verzichtet, lediglich die Zeilen 1350 bis 1450 sind interessant, da hier die Routine zur Umwandlung von ASCII-Zeichen in Punkte zur gleichzeitigen Darstellung von Text und Grafik steht. Diese Routine ist aber leicht verständlich.

Um normal hohe Zeichen zu erhalten, streicht man einfach die Zeile 1450 und ersetzt in Zeile 1440 den Term 2*J+16 durch J+16. Die 16 gibt die Höhe des Textes auf dem Bildschirm an, soll in Kleinschrift geschrieben werden, so muß zu AD 2048 addiert werden.

Sie sehen also, 53248 Bildpunkte sind nicht nur den »größeren« Computern vorbehalten.

Eingabehinweise:
Geben Sie zuerst das Programm »Hyper-Graphics« ein. Dazu steht Ihnen wahlweise Listing 1 oder Listing 2 zur Verfügung. Beachten Sie beim Basic-Lader (Listing 1) bitte die Eingabehinweise auf Seite 76. Nach dem Speichern starten Sie das »Hyper-Graphics« mit RUN. Für Interessierte haben wir in Listing 3 den (nicht einzugebenden) Quellcode abgedruckt. Das Demo-Programm aus Listing 4 laden und starten Sie bitte nur mit aktivierter Erweiterung.

(Joachim Brocke/og)

```
.:2400 BE BE BE A0 88
.:2405 99 90 85 92 AD
.:240A 87 92 81 90 88
.:240F 89 83 93 A0 BC
.:2414 BC BC 3F 54 4F
.:2419 4F 20 4D 41 4E
.:241E 59 20 43 48 41
.:2423 52 41 43 54 45
.:2428 52 53 00 1A 34
.:242D 4E 68 82 9C B6
.:2432 D0 EA 04 1E 38
.:2437 52 6C 86 10 10
.:243C 10 10 10 10 10
.:2441 10 10 10 11 11
.:2446 11 11 11 11 80
.:244B 40 20 10 08 04
.:2450 02 01 09 13 1A
.:2455 21 FF CC 28 43
.:245A 29 20 31 39 38
.:245F 35 20 42 59 20
.:2464 4A 4F 41 43 48
.:2469 49 4D 20 42 52
.:246E 4F 43 4B 45 4C
.:2473 87 24 4C 35 25
.:2478 4C 70 25 4C 1B
.:247D 26 4C 81 26 4C
.:2482 A2 26 4C DC 24
.:2487 AD 05 90 C9 CC
.:248C F0 33 A0 06 B9
.:2491 51 24 99 FF 8F
.:2496 88 D0 F7 A2 D0
.:249B BD FF 1F 9D FF
.:24A0 0F BD CF 20 9D
.:24A5 CF 10 BD 9F 21
.:24AA 9D FF 93 BD 6F
.:24AF 22 9D CF 94 CA
.:24B4 D0 E5 A2 70 BD
.:24B9 3F 23 9D 9F 11
.:24BE CA D0 F7 60 28
.:24C3 03 29 20 31 39
.:24C8 38 35 20 02 19
.:24CD 20 0A 0F 01 03
.:24D2 08 09 0D 20 02
.:24D7 12 0F 03 0B 05
.:24DC AD 05 90 C9 C0
.:24E1 F0 2B A2 D0 BD
.:24E6 FF 0F 9D FF 1F
.:24EB BD CF 10 9D CF
.:24F0 20 BD FF 93 9D
.:24F5 9F 21 BD CF 94
.:24FA 9D 6F 22 CA D0
.:24FF E5 A2 70 BD 9F
.:2504 11 9D 3F 23 CA
.:2509 D0 F7 20 5F E5
.:250E A0 02 A2 16 BD
.:2513 FF 23 9D F9 11
.:2518 98 9D F9 95 CA
.:251D D0 F3 A0 06 B9
.:2522 E3 ED 99 FF 8F
.:2527 88 D0 F7 A9 22
.:252C 8D 01 90 A9 30
.:2531 8D 03 90 60 A9
.:2536 1A A2 D0 9D FF
.:253B 0F 9D CF 10 9D
.:2540 FF 1F 9D CF 20
.:2545 CA D0 F1 A2 60
.:254A A9 00 9D 9F 11
.:254F 9D 9F 21 CA D0
.:2554 F7 A9 12 85 FC
.:2559 A9 00 85 FB A2
.:255E 0E A8 91 FB 88
.:2563 D0 FB E6 FC CA
.:2568 D0 F6 A9 FF 8D
.:256D A8 02 60 20 BF
.:2572 25 20 FD CE 20
.:2577 9E D7 8E AD 02
.:257C 20 EB 25 C9 1A
.:2581 D0 0C AD A8 02
.:2586 C9 1B 90 20 CE
.:258B A8 02 91 FB 20
.:2590 FE 25 11 FD 91
.:2595 FD A5 FC 18 69
.:259A 84 85 FC AC A9
.:259F 02 AD AD 02 91
.:25A4 FB 60 A9 00 85
.:25A9 3E 20 DC 24 A9
.:25AE 16 A0 24 20 1E
.:25B3 CB 4C 62 C4 20
.:25B8 DC 24 A2 0E 4C
.:25BD 37 C4 20 FD CE
.:25C2 20 9E D7 E0 D0
.:25C7 B0 EE 8A 4A 4A
.:25CC 4A 8D A9 02 8A
.:25D1 29 07 8D AA 02
.:25D6 20 FD CE 20 9E
.:25DB D7 8A 4A 4A 4A
.:25E0 4A 8D AB 02 8A
.:25E5 29 0F 8D AC 02
.:25EA 60 AC AB 02 B9
.:25EF 2A 24 85 FB B9
.:25F4 3A 24 85 FC AC
.:25F9 A9 02 B1 FB 60
.:25FE A8 4A 4A 4A 4A
.:2603 18 69 10 85 FE
.:2608 98 0A 0A 0A 0A
.:260D 18 85 FD AD AA
.:2612 02 AA BD 4A 24
.:2617 AC AC 02 60 20
.:261C BF 25 20 EB 25
.:2621 C9 1A F0 5E 8D
.:2626 AE 02 20 FE 25
.:262B 49 FF 31 FD 91
.:2630 FD A0 00 B1 FD
.:2635 D0 49 C8 C0 10
.:263A D0 F7 AC A9 02
.:263F A9 1A 91 FB EE
.:2644 A8 02 A2 D0 BD
.:2649 CF 10 CD A8 02
.:264E F0 11 BD FF 0F
.:2653 F0 03 CA D0 F0
.:2658 AD AE 02 9D FF
.:265D 0F 4C 67 26 AD
.:2662 AE 02 9D CF 10
.:2667 A5 FD 85 FB A5
.:266C FE 85 FC AD A8
.:2671 02 20 FE 25 A0
.:2676 00 B1 FD 91 FB
.:267B C8 C0 10 D0 F7
.:2680 60 20 DC 24 20
.:2685 FD CE 20 D1 E1
.:268A AD A8 02 8D FF
.:268F 23 A2 00 A0 24
.:2694 8A 85 FD A9 12
.:2699 85 FE A9 FD 85
.:269E B9 4C D8 FF 20
.:26A3 DC 24 20 FD CE
.:26A8 20 D1 E1 A9 01
.:26AD 85 B9 A9 00 20
.:26B2 D5 FF AD FF 23
.:26B7 8D A8 02 60 48
.:26BC 8A 48 98 48 AD
.:26C1 1D 91 10 25 2D
.:26C6 1E 91 AA 29 02
.:26CB F0 20 2C 11 91
.:26D0 20 34 F7 20 E1
.:26D5 FF D0 12 20 52
.:26DA FD 20 F0 26 20
.:26DF F9 FD 20 18 E5
.:26E4 20 DC 24 6C 02
.:26E9 C0 4C 56 FF 4C
.:26EE DE FE A9 BB 8D
.:26F3 18 03 A9 26 8D
.:26F8 19 03 60 78 20
.:26FD F0 26 AD 00 03
.:2702 8D 3A 27 AD 01
.:2707 03 8D 3B 27 A9
.:270C 2D 8D 00 03 A9
.:2711 27 8D 01 03 20
.:2716 DC 24 A9 27 85
.:271B 2C 8D 82 02 A9
.:2720 40 85 2B 8D 81
.:2725 02 20 44 C6 58
.:272A 4C 74 C4 8E 39
.:272F 27 20 DC 24 AE
.:2734 39 27 6C 3A 27
```

Listing 2. Hexdump zu »Hyper-Graphics«

```
SPRUNGTABELLE:
.2472 JMP $2487     ZU 'GRAPHIK EINSCHALTEN'
.2475 JMP $2535     ZU 'GRAPHIKSPEICHER LOESCHEN'
.2478 JMP $2570     ZU 'PUNKT SETZEN'
.247B JMP $261B     ZU 'PUNKT LOESCHEN'
.247E JMP $2681     ZU 'GRAPHIK ABSPEICHERN'
.2481 JMP $26A3     ZU 'GRAPHIK LADEN'
.2484 JMP $24DC     ZU 'GRAPHIK AUSSCHALTEN'

GRAPHIK EINSCHALTEN:
.2487 LDA $9005     HOLT ZEIGER AUF CHARACTERGENERATOR
.248A CMP #$CC      PRUEFT AUF 'GRAPHIK ON'-MODUS
.248C BEQ $24C1     SCHON IM GRAPHIKMODUS? DANN ZURUECK INS BASIC
.248E LDY #$06      ZAEHLER FUER ANZAHL DER VIDEOCONTROLLERWERTE
.2490 LDA $2451,Y   HOLT VIDEOCONTROLLERWERTE AUS TABELLE
.2493 STA $8FFF,Y   LAEDT VIDEOCONTROLLER
.2496 DEY           ZAEHLER ERNIEDRIGEN
.2497 BNE $2490     UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.2499 LDX #$D0      ZAEHLER FUER GRAPHIK-KOPIE LADEN
.249B LDA $1FFF,X   BYTE AUS VIDEORAMKOPIE, 1.TEIL, HOLEN
.249E STA $0FFF,X   ...UND INS VIDEORAM BRINGEN
.24A1 LDA $20CF,X   BYTE AUS VIDEORAMKOPIE, 2.TEIL, HOLEN
.24A4 STA $10CF,X   ...UND INS VIDEORAM BRINGEN
.24A7 LDA $219F,X   BYTE AUS FARBRAMKOPIE, 1.TEIL, HOLEN
.24AA STA $93FF,X   ...UND INS FARBRAM BRINGEN
.24AD LDA $226F,X   BYTE AUS FARBRAMKOPIE, 2.TEIL, HOLEN
.24B0 STA $94CF,X   ...UND INS FARBRAM BRINGEN
.24B3 DEX           ZAEHLER ERNIEDRIGEN
.24B4 BNE $249B     UNGLEICH NULL? DANN NAECHSTES ZEICHEN KOPIEREN
.24B6 LDX #$70      ZAEHLER  FUER KOPIE DER ZEICHEN 26-32 DES CHARACTERGENERATORS
.24B8 LDA $233F,X   BYTE AUS KOPIE DIESES CHARACTERGENERATORTEILS HOLEN
.24BB STA $119F,X   ...UND IN CHARACTERGENERATOR EINSETZEN
.24BE DEX           ZAEHLER  ERNIEDRIGEN
.24BF BNE $24B8     UNGLEICH NULL? DANN NAECHSTES ZEICHEN KOPIEREN
.24C1 RTS           ZURUECK INS BASIC

GRAPHIK LOESCHEN:

.2535 LDA #$1A      AKKU MIT CODE FUER GRAPHIKLEERZEICHEN LADEN
.2537 LDX #$D0      ZAEHLER FUER VIDEORAM LADEN
.2539 STA $0FFF,X   1.TEIL DES VIDEORAMS LOESCHEN
.253C STA $10CF,X   2.TEIL DES VIDEORAMS LOESCHEN
.253F STA $1FFF,X   1.TEIL DER VIDEORAMKOPIE LOESCHEN
.2542 STA $20CF,X   2.TEIL DER VIDEORAMKOPIE LOESCHEN
.2545 DEX           ZAEHLER ERNIEDRIGEN
.2546 BNE $2539     UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.2548 LDX #$60      ZAEHLER FUER ZEICHEN 26-32 DES CHARACTERGENERATORS
.254A LDA #$00      AKKU MIT PUNKTFREIEM BYTE LADEN
.254C STA $119F,X   BYTE DER ZEICHEN 26-32 DES CHARACTERGENERATORS LOESCHEN
.254F STA $219F,X   BYTE IN DER KOPIE DIESES TEILS LOESCHEN
.2552 DEX           ZAEHLER ERNIEDRIGEN
.2553 BNE $254C     UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.2555 LDA #$12      AKKU MIT HIGH-BYTE DES CHARACTERGENERATORSTARTS LADEN
.2557 STA $FC       AKKU ALS VEKTOR-HIGHBYTE IN ZEROPAGE BRINGEN
.2559 LDA #$00      AKKU MIT LOW-BYTE DES CHARACTERGENERATORS LADEN
.255B STA $FB       ...UND ALS VEKTOR-LOWBYTE IN ZEROPAGE BRINGEN
.255D LDX #$0E      X-REG. MIT ANZAHL DER PAGES DES CHARACTERGENERATORS LADEN
.255F TAY           Y-REG. ALS ZAEHLER DER BYTES PRO PAGE
.2560 STA ($FB),Y   CHARACTERGENERATORBYTE LOESCHEN
.2562 DEY           ZAEHLER FUER BYTES INNERHALB EINER PAGE ERNIEDRIGEN
.2563 BNE $2560     UNGLEICH NULL? DANN NAECHSTES ZEICHEN LOESCHEN
```

Listing 3. Quell-Code-Listing zu »Hyper-Graphics«

```
.2565 INC $FC        VEKTOR AUF NAECHSTE MEMORYPAGE DES CHARACTERGENERATOR SETZEN
.2567 DEX            ZAEHLER FUER ANZAHL DER ZU BEARBEITENDEN PAGES ERNIEDRIGEN
.2568 BNE $2560      UNGLEICH NULL? DANN NAECHSTE PAGE LOESCHEN
.256A LDA #$FF       CODE FUER OBERSTES FREIES ELEMENT DES CHARACTERGENERATORS
.256C STA $02A8      ...AN FREIEN PLATZ IN PAGE 2 BRINGEN
.256F RTS            ZURUECK INS BASIC

PUNKT SETZEN:
.2570 JSR $25BF      HOLT X- UND Y-KOORDINATE AUS BASICTEXT UND WANDELT UM
.2573 JSR $CEFD      TESTET AUF KOMMA IM BASICTEXT
.2576 JSR $D79E      HOLT BYTEWERT (HIER FARBCODE) AUS BASICTEXT INS X-REGISTER
.2579 STX $02AD      FARBCODE ZWISCHENSPEICHERN
.257C JSR $25EB      HOLT AKTUELL ZU BEARBEITENDES ZEICHEN AUS VIDEORAM
.257F CMP #$1A       PRUEFT AUF LEERZEICHEN
.2581 BNE $258F      KEIN LEERZEICHEN? DANN KEIN NEUES ZEICHEN SETZEN
.2583 LDA $02A8      CODE FUER OBERSTES FREIES ZEICHEN HOLEN
.2586 CMP #$1B       VERGLEICHT MIT CODE FUER LEERZEICHEN +1
.2588 BCC $25AA      KLEINER? DANN FEHLERMELDUNG 'TOO MANY CHARACTERS' AUSGEBEN
.258A DEC $02A8      ZEIGER AUF OBERSTES FREIES ELEMENT DES CHARACTERGEN. ERNIEDR.
.258D STA ($FB),Y    NEUES ZEICHEN INS VIDEORAM BRINGEN
.258F JSR $25FE      BERECHNET VEKTOR AUF CHARACTERGENERATOR & HOLT AKTUELLES BYTE
.2592 ORA ($FD),Y    SETZT PUNKT IM AKTUELLEN BYTE
.2594 STA ($FD),Y    SCHREIBT BEARBEITETES BYTE IN CHARACTERGENERATOR
.2596 LDA $FC        HOLT HIGH-BYTE DES VIDEORAMVEKTORS
.2598 CLC            ADDITION VORBEREITEN
.2599 ADC #$84       WANDELT VEKTOR AUF VIDEORAM IN VEKTOR AUF FARBRAM UM
.259B STA $FC        SETZT HIGH-BYTE DES VEKTORS AUF DAS FARBRAM
.259D LDY $02A9      HOLT BILDSCHIRMSPALTE
.25A0 LDA $02AD      HOLT FARBCODE
.25A3 STA ($FB),Y    BRINGT FARBCODE INS FARBRAM
.25A5 RTS            ZURUECK INS BASIC

FEHLERMELDUNG 'TOO MANY CHARACTERS' AUSGEBEN:
.25A6 LDA #$00       FLAG FUER 'CONT GESPERRT'
.25A8 STA $3E        CONT SPERREN
.25AA JSR $24DC      GRAPHIK AUSSCHALTEN
.25AD LDA #$16       ZEIGER AUF ASCII-TEXT (LOW-BYTE) DES FEHLERTEXTES
.25AF LDY #$24       ZEIGER AUF ASCII-TEXT (HIGH-BYTE) DES FEHLERTEXTES
.25B1 JSR $CB1E      ASCII-TEXT AUSGEBEN
.25B4 JMP $C462      'ERROR' AUSGEBEN + READY-MELDUNG

FEHLERMELDUNG 'ILLEGAL QUANTITY ERROR' AUSGEBEN:
.25B7 JSR $24DC      GRAPHIK AUSSCHALTEN
.25BA LDX #$0E       CODENUMMER FUER 'ILLEGAL QUANTITY' HOLEN
.25BC JMP $C437      FEHLERMELDUNG AUSGEBEN

X- & Y-KOORDINATEN AUS BASICTEXT HOLEN:
.25BF JSR $CEFD      TESTET AUF KOMMA IM BASICTEXT
.25C2 JSR $D79E      HOLT BYTE-WERT (HIER X-KOORDINATE) AUS BASICTEXT
.25C5 CPX #$D0       VERGLEICHT MIT MAXIMALWERT +1
.25C7 BCS $25B7      GROESSER/GLEICH? DANN ZU 'ILLEGAL QUANTITY ERROR' AUSGENBEN
.25C9 TXA
.25CA LSR            ┐
.25CB LSR            ├ X-KOORDINATE/8 BERECHNEN
.25CC LSR            ┘
.25CD STA $02A9      ...UND ZWISCHENSPEICHERN
.25D0 TXA
.25D1 AND #$07       REST VON (X-KOORDINATE/8) BERECHNEN
.25D3 STA $02AA      ...UND ZWISCHENSPEICHERN
.25D6 JSR $CEFD      TESTED AUF KOMMA IM BASICTEXT
.25D9 JSR $D79E      HOLT BYTE-WERT (HIER Y-KOORDINATE) AUS BASICTEXT INS X-REG.
```

Listing 3. Quell-Code-Listing zu »Hyper-Graphics« (Fortsetzung)

```
.25DC TXA
.25DD LSR              ┐
.25DE LSR              |- Y-KOORDINATE/16 BERECHNEN
.25DF LSR              |
.25E0 LSR              ┘
.25E1 STA $02AB        ... UND ZWISCHENSPEICHERN
.25E4 TXA
.25E5 AND #$0F         REST VON Y-KOORDINATE/16 BERECHNEN
.25E7 STA $02AC        ...UND ZWISCHENSPEICHERN
.25EA RTS              ZURUECK ZUR PUNKTSETZ- BZW. -LOESCHROUTINE

AKTUELL ZU BEARBEITENDES ZEICHEN AUS VIDEORAM HOLEN:
.25EB LDY $02AB        BILDSCHIRMZEILE HOLEN
.25EE LDA $242A,Y      VIDEORAMVEKTOR (LOW-BYTE) AUS TABELLE HOLEN
.25F1 STA $FB          ...UND IN ZEROPAGE BRINGEN
.25F3 LDA $243A,Y      VIDEORAMVEKTOR (HIGH-BYTE) AUS TABELLE HOLEN
.25F6 STA $FC          ...UND IN ZEROPAGE BRINGEN
.25F8 LDY $02A9        HOLT BILDSCHIRMSPALTE
.25FB LDA ($FB),Y      HOLT AKTUELLES ZEICHEN AUS VIDEORAM
.25FD RTS              ZURUECK ZUR HAUPTROUTINE

VEKTOR AUF CHARACTERGENERATOR UND PUNKT BERECHNEN:
.25FE TAY              AKTUELLES ZEICHEN ZWISCHENSPEICHERN
.25FF LSR              ┐
.2600 LSR              |-HIGH-BYTE DES VEKTORS AUF DEN CHARACTERGENERATOR BERECHNEN
.2601 LSR              |
.2602 LSR              ┘
.2603 CLC              ADDITION VORBEREITEN
.2604 ADC #$10         OFFSET FUER ANFANG DES CHARACTERGENERATORS ADDIEREN
.2606 STA $FE          HIGH-BYTE DES VEKTORS AUF DEN CHARACTERGENERATORS SETZEN
.2608 TYA              AKTUELLES ZEICHEN HOLEN
.2609 ASL              ┐
.260A ASL              |-LOW-BYTE DES VEKTORS AUF DEN CHARACTERGENERATOR BERECHNEN
.260B ASL              |
.260C ASL              ┘
.260D CLC
.260E STA $FB          LOW-BYTE DES VEKTORS AUF DEN CHARACTERGENERATOR SETZEN
.2610 LDA $02AA        HOLT PUNKTNUMMER
.2613 TAX              IN INDEX WANDELN
.2614 LDA $244A,X      HOLT BIT=PUNKT AUS TABELLE
.2617 LDY $02AC        ZEILE DES AKTUELLEN ZEICHENS HOLEN
.261A RTS              ZURUECK ZUR HAUPTROUTINE

PUNKT LOESCHEN:
.261B JSR $25BF        HOLT X- UND Y-KOORDINATE AUS BASICTEXT
.261E JSR $25EB        HOLT AKTUELLES ZEICHEN AUS VIDEORAM
.2621 CMP #$1A         VERGLEICHT MIT LEERZEICHEN
.2623 BEQ $2680        GLEICH? DANN FERTIG!
.2625 STA $02AE        AKTUELLES ZEICHEN RETTEN
.2628 JSR $25FE        BERECHNET VEKTOR AUF CHARACTERGENERATOR UND HOLT PUNKT
.262B EOR #$FF         BYTE INVERTIEREN
.262D AND ($FD),Y      BIT=PUNKT LOESCHEN
.262F STA ($FD),Y      BYTE IN CHARACTERGENERATOR EINSETZEN
.2631 LDY #$00         INDEX AUF ANFANG DES AKTUELLEN CHARACTERS
.2633 LDA ($FD),Y      EIN BYTE DES AKTUELLEN ZEICHENS HOLEN
.2635 BNE $2680        UNGLEICH NULL (NICHT ALLE PUNKTE GELOESCHT)? DANN FERTIG
.2637 INY              INDEX ERHOEHEN
.2638 CPY #$10         ZEICHEN GANZ DURCHSUCHT?
.263A BNE $2633        NEIN, DANN NAECHSTES BYTE TESTEN
.263C LDY $02A9        BILDSCHIRMSPALTE HOLEN
.263F LDA #$1A         CODE FUER LEERZEICHEN
```

Listing 3. Quell-Code-Listing zu »Hyper-Graphics« (Fortsetzung)

```
.2641 STA ($FB),Y  LEERZEICHEN IN VIDEORAM BRINGEN
.2643 INC $02A8    EIN ZEICHEN FREIGEBEN
.2646 LDX #$D0     ZAEHLER FUER BYTES PRO VIDEORAMHAELFTEN INITIALISIEREN
.2648 LDA $02AE    ZEICHEN AUS ZWISCHENSPEICHER HOLEN
.264B CMP $10CF,X  GLEICH ZEICHEN IN VIDEORAM (2.TEIL) ?
.264E BEQ $2661    JA,DANN ZU 'ZEICHEN VERTAUSCHEN' FUER 2.VIDEORAMTEIL
.2650 CMP $0FFF,X  GLEICH ZEICHEN IN VIDEORAM (1.TEIL) ?
.2653 BEQ $2658    GLEICH, DANN ZU 'ZEICHEN VERTAUSCHEN' FUER 1.VIDEORAMTEIL
.2655 DEX          ZAEHLER ERNIEDRIGEN
.2656 BNE $2648    UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.2658 LDA $02AE    ZEICHEN HOLEN
.265B STA $0FFF,X  ...UND INS VIDEORAM BRINGEN
.265E JMP $2667    ZU 'ZEICHEN KOPIEREN'
.2661 LDA $02AE    ZEICHEN HOLEN
.2664 STA $10CF,X  ...UND INS VIDEORAM BRINGEN
.2667 LDA $FD      ┐
.2669 STA $FB      ├ VEKTOR AUF CHARACTERGENERATOR KOPIEREN
.266B LDA $FE      |
.266D STA $FC      ┘
.266F LDA $02A8    FREIGEGEBENES ZEICHEN HOLEN
.2672 JSR $25FE    VEKTOR AUF CHARACTERGENERATOR ZU DIESEM ZEICHEN BERECHNEN
.2675 LDY #$00     INDEX AUF ANFANG DER ZEICHEN
.2677 LDA ($FD),Y  BYTE DES ALTEN CHARACTERS HOLEN
.2679 STA ($FB),Y  ...UND IN FREIGEWORDENES ZEICHEN EINSETZEN
.267B INY          ZAEHLER ERHOEHEN
.267C CMP #$10     MIT 16 VERGLEICHEN
.267E BNE $2677    UNGLEICH? DANN NAECHSTES BYTE KOPIEREN
.2680 RTS          ZURUECK INS BASIC

GRAPHIK ABSPEICHERN:
.2681 JSR $24DC    GRAPHIK AUSSCHALTEN
.2684 JSR $CEFD    PRUEFT AUF KOMMA IM BASICTEXT
.2687 JSR $E1D1    HOLT FILENAME UND DEVICENUMBER
.268A LDA $02A8    ZEIGER AUF OBERSTES FREIES ZEICHEN HOLEN
.268D STA $23FF    ...UND IN ABZUSPEICHERNDEN RAM-BEREICH BRINGEN
.2690 LDX #$00     LOW-BYTE DER ENDADRESSE DES GRAPHIKSPEICHERS
.2692 LDY #$24     HIGH-BYTE DER ENDADRESSE DES GRAPHIKSPEICHERS
.2694 TXA          LOW-BYTE DER ANFANGSADRESSE DES GRAPHIKSPEICHERS
.2695 STA $FD      ALS VEKTOR-LOW-BYTE IN ZEROPAGE BRINGEN
.2697 LDA #$12     HIGH-BYTE DER STARTADRESSE DES GRAPHIKSPEICHERS HOLEN
.2699 STA $FE      ...UND ALS VEKTOR-HIGH-BYTE IN ZEROPAGE BRINGEN
.269B LDA #$FD     ADRESSE DES VEKTORS AUF START DES GRAPHIKSPEICHERS
.269D STA $B9      ...FUER SAVE-ROUTINE ZUGAENGLICH MACHEN
.269F JMP $FFD8    ZUR 'SAVE'-ROUTINE DES BETRIEBSSYSTEMS

GRAPHIK EINLADEN:
.26A2 JSR $24DC    GRAPHIK AUSSCHALTEN
.26A5 JSR $CEFD    PRUEFT AUF KOMMA IM BASICTEXT
.26A8 JSR $E1D1    HOLT FILENAME UND DEVICENUMBER
.26AB LDA #$01     KONSTANTE FUER 'ABSOLUT LADEN'
.26AD STA $B9      ... FUER LOAD-ROUTINE SETZEN
.26AF LDA #$00     FLAG FUER LOAD
.26B1 JSR $FFD5    ZUR LOAD-ROUTINE DES BETRIEBSSYSTEMS
.26B4 LDA $23FF    ZEIGER AUF OBERSTES FREIES ZEICHEN HOLEN
.26B7 STA $02A8    ... UND AN DIE VORGESEHENE STELLE BRINGEN
.26BA RTS          ZURUECK INS BASIC

NMI-ROUTINE:
.26BB PHA          AKKU RETTEN.
.26BC TXA          X-REGISTER IN AKKU UEBERTRAGEN
.26BD PHA          X-REGISTER RETTEN
```


Listing 3. Quell-Code-Listing zu »Hyper-Graphics«

```
.26BE TYA            Y-REGISTER IN AKKU UEBERTRAGEN
.26BF PHA            Y-REGISTER RETTEN
.26C0 LDA $911D      AKKU MIT IFR DES VIA1 LADEN
.26C3 BPL $26EA      IRQ-BIT GESETZT? DANN ZURUECK
.26C5 AND $911E      NICHT FREIGEGEBENE INTERRUPTS AUSSIEBEN
.26C8 TAX            ERGEBNIS RETTEN
.26C9 AND #$02       SCHIEBEREGISTERFLAG ISOLIEREN
.26CB BEQ $26ED      ZUR NMI-ROUTINE FUER SERIELLE SCHNITTSTELLE
.26CD BIT $9111
.26D0 JSR $F734      ERHOEHT UHRZEIT UND FRAGT STOP-TASTE AB
.26D3 JSR $FFE1      FRAGT AUCH DIE STOP-TASTE AB (IM BETRIEBSSYTEM SO VORGESEHEN)
.26D6 BNE $26EA      NICHT GDRUECKT? DANN ZURUECK
.26D8 JSR $FD52      BETRIEBSSYSTEMVEKTOREN NEU SETZEN
.26DB JSR $26F0      NMI-VEKTOR NEU SETZEN
.26DE JSR $FD9F      INITIALISIERT I/O-REGISTER
.26E1 JSR $E518      INITIALISIERT VIDEOCONTROLLER UND LOESCHT BILDSCHIRM
.26E4 JSR $24DC      SCHALTET AUF HYPERGRAPHIC-TEXTMODUS (= GRAPHIK AUSSCHALTEN)
.26E7 JMP ($C002)    ZUM BASIC-KALTSTART
.26EA JMP $FF56      RUECKKEHR VOM INTERRUPT
.26ED JMP $FEDE      ZUR NMI-ROUTINE FUER SERIELLE SCHNITTSTELLE
.26F0 LDA #$BB       LOW-BYTE DES HYPERGRAPHIC-NMI-VEKTORS HOLEN
.26F2 STA $0318      ...UND EINSETZEN
.26F5 LDA #$26       HIGH-BYTE DES HYPERGRAPHIC-NMI-VEKTORS HOLEN
.26F7 STA $0319      ...UND SETZEN
.26FA RTS            RUECKSPRUNG

HYPERGRAPHICS START:
.26FB SEI            INTERRUPT SPERREN
.26FC JSR $26F0      HYPERGRAPHIC-NMI-VEKTOR SETZEN
.26FF LDA $0300      LOWBYTE DES VEKTORS AUF AUSGABE EINER FEHLERMELDUNG HOLEN
.2702 STA $273A      ... UND RETTEN
.2705 LDA $0301      DAZUGEHOERIGES HIGH-BYTE HOLEN
.2708 STA $273B      ... UND ENTSPRECHEND SETZEN
.270B LDA #$2D       LOW-BYTE DES NEUEN FEHLERMELDUNGSAUSGABEVEKTORS HOLEN
.270D STA $0300      ... UND SETZEN
.2710 LDA #$27       ENTSPRECHENDES HIGH-BYTE HOLEN
.2712 STA $0301      ...UND ENTSPRECHEND SETZEN
.2715 JSR $24DC      HYPERGRAPHIC-MODUS AUF TEXT STELLEN (= GRAPHIK AUSSCHALTEN)
.2718 LDA #$27       HIGH-BYTE DES HYPERGRAPHICS-BASIC-STARTS HOLEN
.271A STA $2C        ... UND IN DIE DAFUER VORGESEHENEN SPEICHERSTELLEN BRINGEN
.271C STA $0282
.271F LDA #$40       ENTSPRECHEND MIT DEM LOW-BYTE VERFAHREN
.2721 STA $FB                    "
.2723 STA $0281                  "
.2726 JSR $C644      BASIC-BEFEHL NEW AUSFUEHREN
.2729 CLI            IRQ WIEDER FREIGEBEN
.272A JMP $C474      ZUM READY-EINSPRUNG

AUSGABE EINER FEHLERMELDUNG:
.272D STX $2739      FEHLERNUMMER RETTEN
.2730 JSR $24DC      GRAPHIK AUSSCHALTEN
.2733 LDX $2739      FEHLERNUMMER HOLEN
.2736 JMP ($273A)    ZU 'FEHLERMELDUNG AUSGEBEN'

GRAPHIK AUSSCHALTEN:
.24DC LDA $9005      HOLT ZEIGER AUF CHARACTERGENERATOR & VIDEORAM
.24DF CMP #$C0       GRAPHIK AUS?
.24E1 BEQ $2510      JA, DANN FERTIG
.24E3 LDX #$D0       ┐
.24E5 LDA $0FFF,X    |
.24E8 STA $1FFF,X    |
```

Listing 3. Quell-Code-Listing zu »Hyper-Graphics« (Fortsetzung)

```
.24EB LDA $10CF,X   |
.24EE STA $20CF,X   |- SIEHE $2499 - $24B4
.24F1 LDA $93FF,X   |  (HIER UMGEKEHRTE FUNKTION)
.24F4 STA $219F,X   |
.24F7 LDA $94CF,X   |
.24FA STA $226F,X   |
.24FD DEX           |
.24FE BNE $24E5    ┘
.2500 LDA ##$70    ┐
.2502 LDA $119F,X   |
.2505 STA $233F,X   |- SIEHE $24B6 - $24BF
.2508 DEX           |  (HIER UMGEKEHRTE FUNKTION)
.2509 BNE $2502    ┘
.250B JSR $E55F     BILDSCHIRM LOESCHEN
.250E LDY ##$02     CODE FUER FARBE ROT IM FARBRAM
.2510 LDX ##$16     ANZAHL DER ZEICHEN PRO ZEILE
.2512 LDA $23FF,X   HOLT BILDSCHIRMCODE AUS TABELLE
.2515 STA $11F9,X   BRING ZEICHEN AUF BILDSCHIRM
.2518 TYA           HOLT FARBCODE
.2519 STA $95F9,X   SETZT FARBCODE
.251C DEX           ZAEHLER ERNIEDRIGEN
.251D BNE $2512     UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.251F LDY ##$06     ZAEHLER FUER ANZAHL DER VIDEOCONTROLLER-WERTE
.2521 LDA $EDE3,Y   HOLT VIDEOCONTROLLER-WERTE AUS DEM ROM
.2524 STA $8FFF,Y   SETZT VIDEOCONTROLLER-WERTE
.2527 DEX           ZAEHLER ERNIEDRIGEN
.2528 BNE $2521     UNGLEICH NULL? DANN VORGANG WIEDERHOLEN
.252A LDA ##$22     BILDSCHIRM VERTIKAL ZENTRIEREN
.252C STA $9001                    "
.252F LDA ##$30     KONSTANTE FUER 24-ZEILEN-BILDSCHIRM HOLEN
.2531 STA $9003     ... UND IN VIDEOCONTROLLER SETZEN
.2534 RTS           ZURUECK INS BASIC
```

Listing 3. Quell-Code-Listing zu »Hyper-Graphics« (Schluß)

```
10 REM*******************************           <064>
20 REM*      HYPER-GRAPHIK DEMO        *        <194>
30 REM* (C) 1985 BY JOACHIM BROCKE *            <026>
40 REM* DIETRICH-BONHOEFFERSTR.-5   *           <013>
50 REM* 4790 PADERBORN              *           <104>
60 REM*******************************           <114>
70 REM FESTLEGUNG DER STARTADRESSEN DER EINZELNEN ROUTINEN   <254>
80 IN=9330:REM'GRAPHIC ON'                      <024>
90 CL=IN+3:REM'GRAPHIC CLEAR'                   <102>
100 SE=IN+6:REM'POINT SET'                      <194>
110 RS=IN+9:REM'RESET POINT'                    <166>
120 GS=IN+12:REM'GSAVE'                         <174>
130 GL=IN+15:REM'GLOAD'                         <181>
140 OF=IN+18:REM'GRAPHIC OFF'                   <201>
150 REM MENUE                                   <082>
160 PRINT"{CLR}HYPER-GRAPHICS DEMO "            <082>
170 PRINT"{DOWN}BITTE WAEHLEN:"                 <151>
180 PRINT"{DOWN}3D-EFFEKT-BILD (-1-)"           <051>
190 PRINT"{DOWN}SINUSKURVE{5SPACE}(-2-)"        <255>
200 PRINT"{DOWN}RECHTECKE{6SPACE}(-3-)"         <233>
210 PRINT"{DOWN}TEXT + GRAPHIK (-4-)"           <253>
220 GET X$                                      <102>
230 IF VAL(X$)<1 OR VAL(X$)>4 THEN 220          <012>
```

Listing 4. Demo-Programm für »Hyper-Graphics«

```
240 REM WAHL DER FARBEN                                                    <043>
250 PRINT"{DOWN}NACH DEM PLOTTEN{6SPACE}TASTE DRUECKEN"                   <162>
260 INPUT"RAHMENFARBE";RF                                                  <253>
270 IF RF<0 OR RF>7 THEN PRINT"FEHLER!":GOTO 260                           <039>
280 INPUT"HINTERGRUNDFARBE";HF                                             <107>
290 IF HF<0 OR HF>15 THEN PRINT"FEHLER!":GOTO 280                          <214>
300 INPUT"PUNKTFARBE";PF                                                   <128>
310 IF PF<0 OR PF>7 OR PF=HF THEN PRINT"FEHLER!":GOTO 300                  <057>
320 POKE 646,PF:REM ZEICHENFARBE IM TEXT-MODUS FESTLEGEN                   <158>
330 POKE 36879,8+RF+16*HF:REM RAHMEN- UND HINTERGRUNDFARBE SETZEN          <037>
340 REM VERZWEIGENINGEWAEHLTESUNTERPROGRAMM:                               <090>
350 ON VAL(X$)GOSUB 1200,400,470,1310:                                     <218>
360 POKE 198,0:WAIT 198,1:REM AUF TASTE WARTEN                             <120>
370 SYS OF:REM GRAPHIK AUSSCHALTEN                                         <209>
380 RUN:REM NEUSTART                                                       <012>
390 REM*** DIE UNTERPROGRAMME ***                                          <233>
400 REM SINUSKURVE                                                         <009>
410 SYS IN:SYS CL                                                          <031>
420 X=-1:FOR I=-↑TO↑STEP 2*↑/208                                           <237>
430 X=X+1:Y=SIN(I)                                                         <159>
440 SYS SE,X,Y*127+127,PF                                                  <009>
450 NEXT I                                                                 <024>
460 RETURN                                                                 <008>
470 REM RECHTECKE                                                          <194>
480 SYS IN:SYS CL                                                          <101>
490 FOR I=16 TO 208-16                                                     <020>
500 SYS SE,I,16,PF                                                         <126>
510 SYS SE,I,255-16,PF                                                     <116>
520 NEXT                                                                   <022>
530 FOR I=31 TO 208-31                                                     <189>
540 SYS SE,I,31,PF                                                         <160>
550 SYS SE,I,255-31,PF                                                     <030>
560 NEXT                                                                   <062>
570 FOR I=16 TO 255-16                                                     <100>
580 SYS SE,16,I,PF                                                         <240>
590 SYS SE,208-16,I,PF                                                     <075>
600 NEXT                                                                   <102>
610 FOR I=31 TO 255-31                                                     <011>
620 SYS SE,31,I,PF                                                         <021>
630 SYS SE,208-31,I,PF                                                     <083>
640 NEXT                                                                   <142>
650 FOR I=64 TO 208-64                                                     <230>
660 SYS SE,I,64,PF                                                         <033>
670 SYS SE,I,255-64,PF                                                     <038>
680 NEXT                                                                   <182>
690 FOR I=71 TO 208-71                                                     <029>
700 SYS SE,I,71,PF                                                         <068>
710 SYS SE,I,255-71,PF                                                     <254>
720 NEXT                                                                   <222>
730 FOR I=86 TO 255-86                                                     <085>
740 SYS SE,86,I,PF                                                         <082>
750 SYS SE,208-86,I,PF                                                     <007>
760 NEXT                                                                   <006>
770 FOR I=90 TO 255-90                                                     <110>
780 SYS SE,90,I,PF                                                         <185>
790 SYS SE,208-90,I,PF                                                     <005>
800 NEXT                                                                   <008>
810 FOR I=86 TO 208-86                                                     <169>
820 SYS SE,I,86,PF                                                         <201>
830 SYS SE,I,255-86,PF                                                     <040>
840 NEXT                                                                   <088>
850 FOR I=90 TO 208-90                                                     <192>
860 SYS SE,I,90,PF                                                         <230>
```

Listing 4. Demo-Programm für »Hyper-Graphics« (Fortsetzung)

```
870 SYS SE,I,255-90,PF                                          <160>
880 NEXT                                                        <128>
890 FOR I=64 TO 255-64                                          <214>
900 SYS SE,64,I,PF                                              <114>
910 SYS SE,208-64,I,PF                                          <145>
920 NEXT                                                        <168>
930 FOR I=71 TO 255-71                                          <013>
940 SYS SE,71,I,PF                                              <088>
950 SYS SE,208-71,I,PF                                          <165>
960 NEXT                                                        <208>
970 FOR I=100 TO 208-100                                        <034>
980 SYS SE,I,100,PF                                             <174>
990 SYS SE,I,255-100,PF                                         <029>
1000 NEXT                                                       <248>
1010 FOR I=102 TO 208-102                                       <076>
1020 SYS SE,I,102,PF                                            <222>
1030 SYS SE,I,255-102,PF                                        <199>
1040 NEXT                                                       <034>
1050 FOR I=100 TO 255-100                                       <112>
1060 SYS SE,100,I,PF                                            <088>
1070 SYS SE,208-100,I,PF                                        <103>
1080 NEXT                                                       <074>
1090 FOR I=102 TO 255-102                                       <154>
1100 SYS SE,102,I,PF                                            <130>
1110 SYS SE,208-102,I,PF                                        <175>
1120 NEXT                                                       <114>
1130 X=-1:FOR I=1 TO 102                                        <210>
1140 Y=I                                                        <010>
1150 SYS SE,I,Y,PF                                              <255>
1160 SYS SE,208-I,Y,PF                                          <135>
1170 SYS SE,I,255-Y,PF                                          <013>
1180 SYS SE,208-I,255-Y,PF                                      <089>
1190 NEXT I:RETURN                                              <118>
1200 REM 3-EFFEKT-BILD                                          <212>
1210 SYS IN:SYS CL                                              <069>
1220 DEF FN A(Z)=90*EXP(-Z*Z/1500)                              <110>
1230 G=100:K=50:FOR X=-103 TO 0:L=-50:H=5*INT(SQR(10816-X*X)/5) <176>
1240 FOR Y=H TO-H STEP-5:Z=25+FN A(SQR(X*X+Y*Y))-.6*Y           <060>
1250 X1=G+X:Y1=256-Z-K:X2=G-X                                   <081>
1260 IF Z>L THEN L=Z:SYS SE,X1+3,Y1,6:SYS SE,X2+3,Y1,6          <046>
1270 NEXT Y,X                                                   <159>
1280 RETURN                                                     <068>
1290 SYS OF:POKE 36879,27:PRINT"{BLUE}                          <199>
1300 RETURN                                                     <088>
1310 REM TEXT+GRAPHIK                                           <002>
1320 INPUT"{CLR}ASCII-TEXT{3SPACE}OKTAEDER{10LEFT}";TE$         <045>
1330 IF LEN(TE$)>16 OR TE$=""THEN 1320                          <200>
1340 SYS CL:SYS IN                                              <126>
1350 REM ASCII-TEXT (A-Z UND SPACE BIS ?)IN PUNKTE WANDELN      <207>
1360 FOR I=1 TO LEN(TE$)                                        <157>
1370 A=ASC(MID$(TE$,I,1)):IF A>63 THEN A=A-64                   <067>
1380 X=I*8:REM STARTKOORDINATE DES TEXTES (X-RICHTUNG)          <006>
1390 AD=32768+A*8:REM ADRESSE DES ZEICHENS IM CHARACTERGENERATOR <033>
1400 FOR J=0 TO 7                                               <207>
1410 A=PEEK(AD+J):REM EINES DER 8 BYTES DES ZEICHENS HOLEN      <150>
1420 FOR K=0 TO 7                                               <235>
1430 P=A AND 2↑(7-K):REM EINES DER 8 BITS PRO ZEILE DES ZEICHENS ISOLIER
     EN                                                         <248>
1440 IF P THEN SYS SE,X+K,2*J+16,PF:REM BIT GESETZT?DANN PUNKT SETZEN <204>
1450 IF P THEN SYS SE,X+K,2*J+17,PF:REM BIT GESETZT?DANN PUNKT SETZEN <218>
1460 NEXT K,J,I                                                 <220>
1470 SYS IN:GOSUB 1490:REM OKTAEDER ZEICHNEN                    <075>
1480 RETURN                                                     <012>
```

Listing 4. Demo-Programm für »Hyper-Graphics« (Fortsetzung)

```
1490 REM OKTAEDER                                  <219>
1500 FOR X=66 TO 133                               <225>
1510 SYS SE,X+30,127,PF                            <150>
1520 SYS SE,X+30,128,PF                            <193>
1530 SYS SE,(X AND 252)+68,127-18,PF               <030>
1540 NEXT X                                        <220>
1550 Y=127:FOR X=67 TO 101                         <159>
1560 SYS SE,(X+30)AND 254,Y,PF                     <123>
1570 SYS SE,X+97,Y,PF                              <004>
1580 SYS SE,X+96,Y,PF                              <012>
1590 Y=Y-.5                                        <107>
1600 NEXT X                                        <024>
1610 REM SYSSE,118,118,1                           <148>
1620 Y=128:FOR X=66 TO 118                         <157>
1630 SYS SE,X+30,Y,PF                              <044>
1640 SYS SE,X+30,Y+1,PF                            <034>
1650 SYS SE,X+30,256-Y,PF                          <062>
1660 SYS SE,X+30,255-Y,PF                          <040>
1670 Y=Y+1.5                                       <014>
1680 NEXT X                                        <104>
1690 Y=110:FOR X=102 TO 118                        <049>
1700 SYS SE,X+30,Y,PF                              <114>
1710 Y=Y+6                                         <043>
1720 NEXT X                                        <144>
1730 Y=207:FOR X=118 TO 133                        <211>
1740 SYS SE,X+30,Y-1,PF                            <150>
1750 SYS SE,X+30,Y-4,PF                            <000>
1760 SYS SE,X+30,Y-2,PF                            <202>
1770 SYS SE,X+30,Y-3,PF                            <244>
1780 SYS SE,X+30,Y-5,PF                            <062>
1790 Y=Y-5                                         <091>
1800 NEXT X                                        <226>
1810 Y=206:FOR X=118 TO 169                        <244>
1820 SYS SE,X+30,Y,PF                              <236>
1830 SYS SE,X+30,Y-1,PF                            <242>
1840 Y=Y-1.9                                       <222>
1850 NEXT X                                        <020>
1860 Y=110:FOR X=102 TO 118                        <221>
1870 SYS SE,X+30,Y,PF                              <030>
1880 Y=Y-3.8                                       <133>
1890 NEXT X                                        <060>
1900 Y=49:FOR X=148 TO 164                         <000>
1910 SYS SE,X,Y,PF                                 <193>
1920 SYS SE,X,Y+1,PF                               <071>
1930 SYS SE,X,Y+2,PF                               <085>
1940 SYS SE,X,Y+3,PF                               <099>
1950 SYS SE,X,Y+4,PF                               <113>
1960 Y=Y+4.8                                       <246>
1970 NEXT X                                        <140>
1980 Y=49:FOR X=148 TO 200                         <137>
1990 SYS SE,X,Y,PF                                 <017>
2000 Y=Y+1.15                                      <192>
2010 NEXT X                                        <180>
2020 RETURN                                        <044>
```


Listing 4. Demo-Programm für »Hyper-Graphics« (Schluß)

19 tolle Grafik-Befehle für den VC 20

Mit dieser Basic-Erweiterung kommen auch Sie als VC 20-Besitzer mit einer 16-KByte-Erweiterung in den Genuß, hochauflösende Grafiken einfach zu erstellen.

Für die Befehlserweiterung ist mindestens eine Speichererweiterung von 16 KByte erforderlich. Das normale Commodore-Basic V.2 wird um 19 Befehle und 5 Funktionen erweitert. 13 dieser neuen Befehle dienen zur leichteren Handhabung der Grafikfähigkeiten des VC 20.

Mit GLOAD und GSAVE können die Grafiken dann geladen und gespeichert werden.

Es besteht die Möglichkeit, Grafiken anderer Programme zu laden und auf den Bildschirm zu bringen. Da jedoch oft das Bildschirmformat unterschiedlich ist, muß der Bildschirm der jeweiligen Grafik angepaßt werden. Dies wird mit Hilfe des SCREEN-Befehls erreicht. Für das Format von »JOYPAINT« aus 64'er, Ausgabe 2/85 hätte der Befehl folgendes Aussehen:

```
SCREEN 21,9,64
```

Das heißt:
21 Spalten (8*21 = 168 Bildpunkte)
9 Zeilen (16*9 = 144 Bildpunkte)
64 = 1. Zeichen in Bildschirmcode links oben in der Bildschirmecke.

Ist der Grafikmodus eingeschaltet, kann kein Text eingeblendet werden. Einige der Grafikbefehle sind der Turtle-Grafik für den C 64 aus der Ausgabe 11/84 entliehen. Zur einfacheren Handhabung des Textmodus sind die Befehle CURSOR und SCROLL gedacht.

Die Speicherplatzbelegung sieht wie folgt aus:

$1000 - $11FF	Bildschirm
$1200 - $1FFF	Grafik
$2000 - $2AFF	Basic-Erweiterung
$2B00 - $XXXX	Basic-Programmspeicher

Als Sprungvektoren werden benutzt:

$0300 / $0301	Vektor für Fehlermeldungen
$0304 / $0305	Vektor für Interpretercode
$0306 / $0307	Vektor für LIST
$0308 / $0309	Vektor für Basic-Befehl
$030A / $030B	Vektor für Ausdruck auswerten

Der Speicherbereich $02A6 bis $02FF wird vom Programm als Zwischenspeicher genutzt.

Die x-Position des Grafik-Cursors ist in $02A6 gespeichert, die y-Position in $02A7.

Eingabe des Programms

Vor der Programmeingabe muß folgende Zeile eingegeben werden

```
1 SYS 8192
```

Dann wird im Direktmodus eingegeben:

```
POKE 44,43:POKE 43*256,0:NEW
```

Hiermit wird der Basic-Anfang auf $2B00 (dez=43*256) hochgesetzt. Jetzt kann der Basic-Lader eingegeben werden. Lief er ohne Fehlermeldung durch, wird POKE 44,16 eingetippt. Läßt man das Programm jetzt listen, so darf nur 1 SYS 8192 auf dem Bildschirm erscheinen.

Jetzt kann das Programm wie jedes Basic-Programm gespeichert werden.

Alle Zusatzbefehle und Funktionen lassen sich wie die Commodore-Befehle abkürzen.

Befehlserläuterungen

CURSOR xp, yp, x$:
xp = x-Koordinate;
yp = y-Koordinate;
x$ = Ausgabestring, er kann den gleichen Aufbau haben wie der PRINT-Befehl.
Kann nur im Textmodus benutzt werden

SCROLL:
Der Bildschirminhalt wird nach oben gescrollt

SOUND tg, th, la:
tg = Tongenerator 0 bis 3
th = Tonhöhe 128 bis 255
la = Lautstärke 0 bis 15

RENEW:
Nach einem Reset oder NEW kann mit RENEW das Programm wieder in den Speicher geholt werden. Es darf jedoch in der Zwischenzeit keine Variable definiert worden sein.

OFF:
Die Fehlermeldungen werden unterdrückt.

NORM:
Die Fehlermeldungen werden wieder eingeschaltet.

HIRES, mo, hi, ra:
mo = Text- oder Grafikmodus (Text : mo = 0; Grafik: mo = 1)
hi = Hintergrundfarbe 0 bis 15
ra = Rahmenfarbe 0 bis 7
bei mo = 1 wird die Zeichenfarbe = 1, der Grafik-Cursor an die Position 80,80 gesetzt.

COLOR fa, hi, ra:
fa =Farbe, in der gezeichnet wird
hi =Hintergrundfarbe
ra =Rahmenfarbe
Die Hintergrund- und Rahmenfarbe kann auch weggelassen werden. Das gleiche gilt für den Befehl HIRES. Die Befehle sähen dann wie folgt aus:
»HIRES mo« oder »COLOR fa«.

SCREEN sp, zl, ze:
sp = Bildschirmbreite;
zl = Bildschirmhöhe;
ze = erstes Zeichen auf dem Grafikschirm
Im normalen Grafikmodus ist sp = 20, zl = 10, ze = 32. Das heißt, es gibt 8*20=160 Punkte in x-Richtung und 16*10=160 Punkte in y-Richtung.
Im Textmodus entspricht sp der Spaltenzahl und zl der Zeilenzahl. ze kann weggelassen werden, wenn nur die Bildgröße verändert werden soll. Mit Hilfe dieses Befehls ist es möglich, Grafiken anderer Grafikprogramme auf den Bildschirm zu bringen.

GLOAD x$, nr:
x$ = Name der zu ladenden Grafik;
nr = Device-Nummer
(Kassette nr = 1; Diskette nr = 8)
Es können auch Grafiken anderer Programme geladen werden.

GSAVE x$, nr:
x$ = Name der zu speichernden Grafik
nr = Gerätenummer

CLEAR:
Grafikbildschirm löschen

MODE mo:
mo= Zeichenmodus
(Zeichen mo = 0; Löschen mo = 1; Invertieren mo = 2)
mo > 2 hebt den Grafik-Cursor, es wird weder gezeichnet noch gelöscht.

PLOT xp, yp:
xp = x-Koordinate;
yp = y-Koordinate
Der Grafik-Cursor wird an die Position xp, yp gesetzt.

DRAW xp, yp:
xp =x-Koordinate;
yp =y-Koordinate
Es wird eine Linie von der alten x, y-Position zu den Koordinaten xP, yP gezogen.

DEG ri:
ri = Richtung

```
    0
  7   1
6   *   2
  5   3
    4
```

RETURN ri:
Die Richtung wird um »ri«-Einheiten nach rechts gedreht.

LTURN ri
Die Richtung wird um »ri«-Einheiten nach links gedreht.

MOVE vo:
Es wird eine Linie mit »vo«-Punkten in die gesetzte Richtung gezogen.

Funktionserläuterungen

JOY:
Ist (JOY AND 23)=16, dann wird der Joystick in keine Richtung bewegt.
Ist (JOY AND 23) < 16, dann wird der Joystick in die angegebene Richtung gedrückt.
Bei (JOY AND 8)=8 wird der Feuerknopf betätigt.

MULT$(x$, xx)
x$ = String;
xx = Wert, mit dem x$ verlängert wird.
Beispiel:
MULT$(" +." ,3) = " +.+.+."

HEX$(xx):
xx wird in einen 4-stelligen Hexadezimalstring umgewandelt.

BIN$(xx):
xx wird in einen 8-stelligen Binärstring umgewandelt.

DEC(x$):
x$ wird in eine Dezimalzahl umgerechnet.
Ist LEN(x$)= 4 oder =2, dann wird vom Hexadezimalsystem ausgegangen.
Ist LEN(x4)=8, wird vom Binärsystem ausgegangen.

$XXXX
$XXXX ist eine 4-stellige Hexadezimalzahl.
Beispiel:
?$1E00
7680

Lassen Sie sich also nicht durch die »endlose« DATA-Wüste abschrecken.
Denn hat man diese Einöde bewältigt, so verfügt man über eine nützliche Befehlserweiterung.

(Th. Schaper/ah)

```
20 REM THORSTEN SCHAPER                      <237>
25 REM KURZE KAMPSTR.21                      <010>
30 REM 3300 BRAUNSCHWEIG                     <068>
35 REM                                       <097>
50 PRINT"(CLR)**** TH.BASIC GR *****"        <001>
55 PRINT"(DOWN)(C) BY                        <028>
60 PRINT"(DOWN,3SPACE)THORSTEN SCHAPER"      <102>
65 PRINT"(DOWN,3SPACE)KURZE KAMPSTR.21"      <200>
70 PRINT"(DOWN,3SPACE)3300 BRAUNSCHWEIG"     <191>
75 PRINT"(2DOWN,2SPACE)BITTE WARTEN"         <155>
100 FOR N=8192 TO 10912 STEP 16              <108>
105 CS=0                                     <065>
110 FOR M=0 TO 15                            <005>
115 READ A                                   <155>
120 CS=CS+A                                  <028>
125 POKE N+M,A                               <075>
130 NEXT M                                   <246>
135 READ CC                                  <243>
140 IF CC<>CS THEN PRINT"DATAERROR IN"1000
    +5*(N-8192)/16:STOP                      <212>
145 NEXT N                                   <013>
150 SYS 8192                                 <151>
1000 DATA 120,234,234,234, 32,141,253,169,
      43,141,130,  2, 32, 82,253, 32,2132  <118>
1005 DATA 249,253, 32, 24,229,162, 11,189,
      59, 32,157,  0,  3,202, 16,247,1865  <221>
1010 DATA 88,234,234,234, 32,164,227,165,
     43,164, 44, 32,  8,196,169, 71,2105   <045>
1015 DATA 160, 32, 32, 30,203, 32, 18,228,
      76,129,227, 58,196,131,196,229,1977  <158>
1020 DATA 32,152, 33,226, 33, 17, 41, 42,
     42, 42, 42, 32, 84, 72, 46, 32, 968   <036>
1025 DATA 66, 65, 83, 73, 67, 32, 71, 82,
     32, 42, 42, 42, 42, 13, 29, 29, 810   <052>
```

```
1030 DATA 29, 29, 29, 71, 82, 65, 70, 73,
     75, 13, 13,  0,  0,  0,  0,  0, 549   <080>
1035 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <066>
1040 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <071>
1045 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <076>
1050 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <081>
1055 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <086>
1060 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <091>
1065 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <096>
1070 DATA  0,  0,  0,  0,  0,166,122,160,
      4,132, 15,189,  0,  2, 16,  7, 813   <011>
1075 DATA 201,255,240, 62,232,208,244,201,
      32,240, 55,133,  8,201, 34,240,2586  <042>
1080 DATA 86, 36, 15,112, 45,201, 63,208,
      4,169,153,208, 37,201, 48,144,1730   <201>
1085 DATA  4,201, 60,144, 29,132,113,160,
      0,132, 11,136,134,122,202,200,1780   <102>
1090 DATA 232,189,  0,  2, 56,249,158,192,
     240,245,201,128,208, 48,  5, 11,2164  <211>
1095 DATA 164,113,232,200,153,251,  1,185,
     251,  1,240, 89, 56,233, 58,240,2467  <120>
1100 DATA  4,201, 73,208,  2,133, 15, 56,2
     33, 85,208,159,133,  8,189,  0,1707   <199>
1105 DATA  2,240,223,197,  8,240,219,200,1
     53,251,  1,232,208,240,166,122,2702   <077>
1110 DATA 230, 11,200,185,157,192, 16,250,
     185,158,192,208,180,160,255,202,2781  <114>
1115 DATA 200,232,189,  0,  2, 56,249,105,
      34,240,245,201,128,208,  2,240,2331  <199>
1120 DATA 173,166,122,230, 11,200,185,104,
      34, 16,250,185,105, 34,208,226,2249  <001>
1125 DATA 189,  0,  2, 16,155, 76,  9,198,
      16, 66,201,255,240, 62, 36, 15,1536  <136>
1130 DATA 48, 58,170,132, 73,201,204,176,
     10,160,192,132, 35,160,158,132,2041   <003>
1135 DATA 34,208, 11,233, 76,170,160, 34,1
     32, 35,160,105,132, 34,160,  0,1684   <069>
1140 DATA 10,240, 16,202, 16, 12,230, 34,2
     08,  2,230, 35,177, 34, 16,246,1708   <205>
1145 DATA 48,241,200,177, 34, 48,  8, 32,
     71,203,208,246, 76,243,198, 76,2109   <214>
1150 DATA 239,198, 32,201, 39,201,204,144,
      25,201,252,176, 21, 32,243, 33,2241  <213>
1155 DATA 76,174,199,233,203, 10,168,185,
      9, 34, 72,185,  8, 34, 72, 76,1738   <167>
1160 DATA 115,  0, 32,121,  0, 76,231,199,
     100, 37,255, 34,145, 35,169, 35,1584  <131>
1165 DATA 183, 35,205, 35,219, 35,230, 35,
     175, 36,216, 36,235, 36, 32, 37,1780  <150>
1170 DATA 52, 37,111, 37, 22, 38, 44, 38,
     85, 38,116,233,118, 38,  7,207,1221   <103>
1175 DATA  7,207,  7,207,  7,207, 47, 41,
      7,207,  7,207,111, 39,212, 39,1559   <007>
1180 DATA 78, 40,174, 40,232, 40,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0, 604   <000>
1185 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0,  0,  0,  0,  0,  0,  0,  0,   0   <216>
1190 DATA  0,  0,  0,  0,  0,  0,  0,  0,
      0, 78, 79, 82,205, 72, 73, 82, 671   <217>
1195 DATA 69,211, 67, 79, 76, 79,210, 77,
     79, 68,197, 67, 76, 69, 65,210,1699   <244>
1200 DATA 82, 84, 85, 82,206, 76, 84, 85,
     82,206, 68, 69,199, 80, 76, 79,1643   <097>
1205 DATA 212, 77, 79, 86,197, 82, 69, 78,
      69,215, 71, 83, 65, 86,197, 71,1737  <122>
1210 DATA 76, 79, 65,196, 79, 70,198, 68,
     82, 65,215, 83, 79, 85, 78,196,1714   <252>
1215 DATA 67, 85, 82, 83, 79,210, 83, 67,
     82, 79, 76,204, 83, 67, 82, 69,1498   <072>
1220 DATA 69,206, 67, 65, 76,204, 83, 85,1
     94, 67, 73, 82, 67, 76,197, 73,1684   <086>
1225 DATA 78, 76, 73, 78,197, 84, 69, 88,2
     12, 77, 85, 76, 84,201, 86, 69,1633   <136>
1230 DATA 82, 83, 73, 79,206, 74, 79,217,
     68, 69,195, 77, 85, 76, 84,164,1711   <077>
1235 DATA 72, 69, 88,164, 66, 73, 78,164,
      0,  0,  0,  0,  0,  0,  0,  0, 774   <222>
1240 DATA 32,158,215,224,  0,208,  6, 32,
     61, 39, 76, 26, 35,224,  1,208,1545   <144>
1245 DATA  6, 32, 70, 39, 76, 26, 35, 76,
     72,210, 32,121,  0,240, 35, 32,1102   <107>
1250 DATA 253,225,142,168,  2, 32,253,225,
     224,  8, 16,235,142,169,  2,173,2269  <229>
1255 DATA 168,  2,201, 16, 16,225, 10, 10,
      10, 10,  9,  8, 13,169,  2,141,1010  <004>
1260 DATA 15,144, 96,169, 22,141,  2,144,1
     69, 46,141,  3,144,169, 38,141,1584   <001>
1265 DATA  1,144,169, 12,141,  0,144,169,1
     92,141,  5,144, 96,169, 20,141,1688   <020>
1270 DATA  2,144,  9,  1,141,  3,144,169,
     45,141,  1,144,169, 14,141,  0,1268   <148>
1275 DATA 144,169,204,141,  5,144,162,  0,
     138, 24,105, 32,157,  0, 16,169,1610  <120>
1280 DATA  1,157,  0,148,232,224,200,208,2
     39,169, 80,141,166,  2,141,167,2275   <031>
1285 DATA  2, 96, 32,158,215,224,  8, 48,
      3, 76, 72,210,138,162,  0,157,1601   <016>
1290 DATA  0,148,232,224,200,208,248, 76,
     26, 35, 32,158,215,224,  4, 48,2078   <042>
1295 DATA  3, 76, 72,210,142,175,  2, 96,1
     62, 18,134,252,160,  0,132,251,1885   <114>
1300 DATA 162, 14,152,145,251,136,208,251,
     230,252,202,208,246, 96, 32,158,2743  <042>
1305 DATA 215,138, 24,109,174,  2, 41,  7,
     141,174,  2, 96, 32,158,215, 76,1604  <110>
1310 DATA 88, 37,234,234,234,234,234, 32,1
     58,215,224,  8, 48,  3, 76, 72,2131   <076>
1315 DATA 210,142,174,  2, 96, 76,255, 38,
     166,  2, 16, 12, 41,127,201, 32,1590  <201>
1320 DATA 48,  6, 32,137, 35, 76, 72,210,2
     02, 16,236,173,166,  2,170, 41,1622   <037>
1325 DATA  7,141,170,  2,138, 74, 74, 74,1
     41,172,  2,173,167,  2,170, 41,1548   <220>
1330 DATA 15,141,171,  2,138, 74, 74, 74,
     74,141,173,  2,168,173,172,  2,1594   <149>
1335 DATA 192,  0,240,  6, 24,101,254,136,
     208,250,170,189,  0, 16,134,254,2174  <179>
1340 DATA 141, 52,  3,140, 53,  3,162,  4,
      24, 14, 53,  3, 14, 52,  3,144, 865  <000>
1345 DATA  3,238, 53,  3,202,208,241,173,
     53,  3, 24,105, 16,133,252,173,1880   <233>
1350 DATA 52,  3,133,251,169,128,174,170,
      2,240,  4,074,202,208,252,133,2195   <051>
1355 DATA 253,172,171,  2,173,175,  2,208,
      7,165,253, 17,251,145,251, 96,2341   <139>
1360 DATA 201,  1,208,  9,165,253, 73,255,
      49,251,145,251, 96,201,  2,208,2368  <114>
1365 DATA  6,177,251, 69,253,145,251, 96,
      0,207,  0,255,  0,255,  0,191,2156   <018>
1370 DATA  0,  1,  1,  1,  0,255,255,255,2
     55,255,  0,  1,  1,  1,  0,255,1536   <080>
1375 DATA 32,158,215,142,166,  2, 32,253,2
     25,142,167,  2, 76,245, 35,174,2066   <173>
1380 DATA 174,  2,189,160, 36, 24,109,166,
      2,141,166,  2,189,168, 36, 24,1588   <213>
1385 DATA 109,167,  2,141,167,  2, 76,245,
      35, 32,158,215,224,  0,240, 11,1824  <215>
1390 DATA 142,176,  2, 32,191, 36,206,176,
      2,208,248, 96,160,  1,152,145,1973   <011>
1395 DATA 43, 32, 51,197,165, 34, 24,105,
      2,133, 45,165, 35,105,  0,133,1269   <079>
1400 DATA 46, 32, 96,198, 76,116,196,169,
      0, 32,189,255,162,  1,160,  1,1729   <179>
1405 DATA 32,186,255, 32,  3,226, 32, 84,2
     26, 32,253,225,160,  1, 76,186,2009   <203>
1410 DATA 255, 32,  7, 37,162,  0,160, 18,
     132,194,134,193,160, 32,132,175,1823  <035>
1415 DATA 134,174, 76,133,246,166, 45,134,
     251,164, 46,132,252,169,  0,133,2255  <184>
1420 DATA 10, 32,  7, 37,165, 10,162,  0,1
     60, 18, 32,213,255,166,251,164,1682   <074>
1425 DATA 252,134, 45,132, 46, 76,117,225,
     173,174,  2,142,174,  2, 56,237,1987  <170>
1430 DATA 174,  2, 76,214, 35,169, 58,141,
      0,  3,169,196,141,  1,  3, 96,1478   <028>
1435 DATA 173,  8,  3,141,  0,  3,173,  9,
      3,141,  1,  3, 96,169,255,141,1319   <111>
1440 DATA 192,  2,170, 56,173,192,  2,237,
     193,  2,176,  3,206,194,  2,141,1941  <071>
1445 DATA 192,  2,232,173,194,  2,208,235,
      96,160,  1,162,  0,185,166,  2,2010  <036>
```

```
1450 DATA 56,249,195,  2,240,  7,176,  4,2
     32, 76,173, 37,202,142,197,  2,1990  <013>
1455 DATA 224,  1,208,  7, 56,185,195,  2,
     249,166,  2,153,199,  2,173,197,2019 <214>
1460 DATA  2,153,197,  2,136, 16,212,173,1
     99,  2,141,193,  2, 56,205,200,1889  <008>
1465 DATA  2,176,  6,173,200,  2,141,193,
      2,160,  1,185,199,  2,141,194,1777  <101>
1470 DATA  2, 32,125, 37,138,153,199,  2,1
     36, 16,240,169,  0,141,208,  2,1600  <102>
1475 DATA 141,209,  2,160,  1,185,208,  2,
      24,121,199,  2,153,208,  2,144,1761 <161>
1480 DATA 10, 24,185,166,  2,121,197,  2,1
     53,166,  2,136, 16,231, 32,245,1688  <143>
1485 DATA 35,206,193,  2,208,221, 96, 32,1
     58,215,142,195,  2, 32,253,225,2215  <097>
1490 DATA 142,196,  2, 76,153, 37,153,199,
       2,238,193,  2, 96, 32,158,215,1894 <083>
1495 DATA 224,  4,144,  3, 76, 72,210,134,
     250, 32,253,225,138,166,250,157,2338 <194>
1500 DATA 10,144, 32,253,225,224, 16,176,2
     35,134,250,173, 14,144, 41,240,2311  <071>
1505 DATA  5,250,141, 14,144, 96, 32,158,2
     15,224, 22,144,  3, 76, 72,210,1806  <196>
1510 DATA 134,250, 32,253,225,224, 23,176,
     244, 32,253,206,164,250, 32, 12,2510 <118>
1515 DATA 229, 32,121,  0, 76,160,202, 32,
     158,215,173,  2,144, 41,128,133,1846 <003>
1520 DATA 251,138, 41,127,  5,251,133,251,
     134,252,169, 34, 56,229,252,133,2456 <002>
1525 DATA 252, 32,253,225,138, 10,133,253,
     173,  3,144, 41,129,  5,253,141,2185 <066>
1530 DATA  3,144,165,252,141,  0,144,165,2
     51,234, 76, 33, 39,169,  0,133,1949  <018>
1535 DATA 13, 32,115,  0,201, 36,208,  3,
     76,200, 38,201,223,144,  3, 76,1569  <068>
1540 DATA 93, 39, 32,121,  0, 76,141,206,
     32,219, 38,133, 98, 32,219, 38,1517  <032>
1545 DATA 133, 99,162,144, 56, 32, 73,220,
      76,115,  0, 32,115,  0, 32,240,1529 <187>
1550 DATA 38, 10, 10, 10, 10,133,251, 32,1
     15,  0, 32,240, 38,  5,251, 96,1271  <201>
1555 DATA 201, 58,144,  2,105,  8, 41, 15,
      96, 32, 95,229, 76, 67, 35,173,1377 <113>
1560 DATA  2,144, 41,127,133,254, 10, 10,
     10,205,166,  2,176,  3, 76, 72,1431  <233>
1565 DATA 210,173,  3,144, 41,126, 10, 10,
      10,205,167,  2,144,240, 76, 11,1572 <170>
1570 DATA 36,141,  2,144, 32,121,  0,201,
     44,208, 17, 32,253,225,134,254,1844  <139>
1575 DATA 162,  0,138, 24,101,254,157,  0,
      16,232,208,246, 96, 32, 95,229,1990 <067>
1580 DATA 32,101, 37, 76, 67, 35,169, 83,1
     41,  0,  3,169, 39,141,  1,  3,1097  <119>
1585 DATA 76, 93, 35,134,254, 32, 61, 39,1
     66,254, 76, 58,196, 56,233,223,1986  <041>
1590 DATA 10,170,189, 46, 34,133,251,189,
     47, 34,133,252,108,251,  0,173,2020  <230>
1595 DATA 34,145, 72,169,127,141, 34,145,1
     74, 32,145,134,251,104,141, 34,1882  <201>
1600 DATA 145,173, 17,145, 41, 60,133,252,
     165,251, 41,128,  5,252,133,251,2192 <078>
1605 DATA 160,  0, 41, 32,208,  2,160,  8,
     132,252,162,  7,165,251, 61,185,1826 <027>
1610 DATA 39,208, 14,189,193, 39,  5,252,1
     68, 32,162,211, 76,115,  0,234,1937  <046>
1615 DATA 234,202, 16,232,169, 16, 76,166,
      39,  4,  8, 16,128, 20, 24,132,1482 <085>
1620 DATA 136,  0,  4,  6,  2,  7,  5,  1,
       3, 32,115,  0,201,230,144,  3, 889 <105>
1625 DATA 76,  8,207, 96, 32, 63, 40, 32,1
     63,214,134,251,132,252,201,  2,1903  <143>
1630 DATA 240, 11,201,  4,240, 19,201,  8,
     240, 38, 76, 72,210,160,  0, 32,1752 <117>
1635 DATA 43, 40,168, 76,162,211, 76,115,
      0,160,  0, 32, 43, 40,133, 98,1397  <200>
1640 DATA 160,  2, 32, 43, 40,133, 99,234,
     234, 56, 76, 69, 40, 76,115,  0,1409 <070>
1645 DATA 160,  0,132,254,177,251,  6,254,
      41,  1,  5,254,133,254,200,192,2314 <025>
1650 DATA  8,208,241,164,254, 76,162,211,
     76,115,  0,177,251, 32,240, 38,2253  <125>
1655 DATA 10, 10, 10, 10,200,133,254,177,2
     51, 32,240, 38,  5,254, 96, 32,1752  <179>
1660 DATA 115,  0, 76,241,206,162,  0,134,
      13,162,144, 76, 73,220, 32,115,1769 <130>
1665 DATA  0, 32,250,206, 32,158,205, 32,1
     63,214,134,251,132,252,133,253,2447  <054>
1670 DATA 32,253,206, 32,158,215,134,254,
     32,247,206,166,254,240, 14,169,2612  <041>
1675 DATA  0, 24,101,253,144,  3, 76, 88,2
     14,202,208,245,170,138, 32,125,2023  <061>
1680 DATA 212,165, 97,240, 34,160,  0,132,
      20,132, 21,164, 21,177,251,164,1990 <105>
1685 DATA 20,145, 98,230, 20,230, 21,165,
     21,197,253,208,  4,169,  0,133,1914  <213>
1690 DATA 21,165, 20,197, 97,208,228,162,2
     55,134, 13, 76,202,212, 32,115,2137  <229>
1695 DATA  0, 32,250,206, 32,138,205, 32,2
     47,215, 32,247,206,169,  4, 32,2047  <232>
1700 DATA 125,212,165, 21,160,  0, 32,209,
      40,165, 20, 32,209, 40, 76,167,1673 <232>
1705 DATA 40, 72, 74, 74, 74, 74, 32,220,
     40,104, 41, 15,201, 10,144,  2,1217  <114>
1710 DATA 105,  6,105, 48,145, 98,200, 96,
      32,115,  0, 32,250,206, 32,158,1628 <041>
1715 DATA 215,134,251, 32,247,206,169,  8,
      32,125,212,160,  0,162, 48,165,2166 <190>
1720 DATA 251, 16,  1,232,138,145, 98,  6,
     251,200,192,  8,208,239, 76,167,2228 <194>
1725 DATA 40,169,  0,133, 13, 32,115,  0,2
     01, 36,208,  3, 76,200, 38,201,1465  <233>
1730 DATA 255,240,  7,201,230,144,  3, 76,
      93, 39, 32,121,  0, 76,141,206,1864 <144>
1735 DATA 32,121,  0,201, 79,240, 22, 32,1
     38,205, 32,247,215,165, 20,141,1890  <051>
1740 DATA 180,  2,165, 21,141,181,  2, 32,
     253,206, 76, 80, 41, 32,115,  0,1527 <049>
1745 DATA 32,158,215,134,211, 32,253,225,1
     34,214, 32,253,206, 32,158,205,2494  <020>
1750 DQTA 32,163,214,134, 98,132, 99,133,
     97,160,  0,177, 98,140,182,  2,1861  <007>
1755 DATA 32,157, 42,172,182,  2,200,196,
     97,208,240, 96,201, 32, 16, 40,1913  <100>
1760 DATA 201, 13,208,  7,169,  0,133,211,
     230,214, 96,201, 17,240,249,201,2390 <026>
1765 DATA 18,208,  3,133,199, 96,201, 19,2
     08,  7,169,  0,133,211,133,214,1952  <119>
1770 DATA 96,201, 29,208,  2,230,211, 96,2
     01, 64, 16,  3, 76,255, 41,201,1930  <049>
1775 DATA 96, 16,  5, 41, 31, 76,255, 41,2
     34,170, 48,  5, 41, 95, 76,255,1485  <171>
1780 DATA 41, 76,131, 42, 37,201,141,240,1
     87,201,145,208,  3,198,214, 96,2161  <147>
1785 DATA 201,146,208,  5,169,  0,133,199,
      96,201,147,208,  6, 32,184, 35,1970 <053>
1790 DATA 76,154; 41,201,157,208,  2,198,2
     11, 96,201,192,176,  6, 56,233,2208  <084>
1795 DATA 64, 76,255, 41,201,255,208,  5,1
     69, 94, 76,255, 41, 41,127,141,2049  <100>
1800 DATA 179,  2,173,166,  2,141,176,  2,
     173,167,  2,141,177,  2,173,175,1851 <151>
1805 DATA  2,141,178,  2,169,  3,141,175,
      2,165,211, 10, 10, 10,141,166,1526  <059>
1810 DATA  2,165,214, 10, 10, 10,141,167,
      2, 32,245, 35,173,176,  2,141,1525  <233>
1815 DATA 166,  2,173,177,  2,141,167,  2,
     173,178,  2,141,175,  2,173,171,1845 <248>
1820 DATA  2, 24,101,251,133,251,165,252,1
     05,  0,133,252,173,179,  2,166,2189  <074>
1825 DATA 199,240,  2,  9,128,133,253,169,
       0,133,254,162,  2,  6,254,  6,1950 <056>
1830 DATA 253,144,  2,230,254,202, 16,245,
     165,253, 24,109,180,  2,133,253,2465 <162>
1835 DATA 165,254,109,181,  2,133,254,160,
       7,177,253,145,251,136, 16,249,2492 <073>
1840 DATA 230,211, 96, 41,127,201,127,208,
       2,169, 94,201, 32,176,  5,  9,1929 <217>
1845 DATA 128, 76,197, 41,201, 64,176,  2,
       9, 64, 76,255, 41,177, 98, 16,1621 <206>
1850 DATA  3, 76,131, 42, 76,124, 41,  0,
       0,  0,  0,  0,  0,  0,  0,  0, 493 <155>
```


Listing zu »19 Grafik-Befehle«. Bitte beachten Sie die Eingabehinweise auf Seite 76.

Checksummer 20 V3

Der Checksummer 20 V3 für den VC 20 überprüft jede Basic-Zeile direkt nach der Eingabe, erkennt Fehleingaben und auch Vertauschungen von Zahlen und Ziffern, und erspart deshalb eine aufwendige Fehlersuche.

Der Checksummer 20 V3 ist ein kleines Maschinenprogramm, das Sie sofort unterrichtet, ob Sie die jeweilige Programmzeile korrekt eingegeben haben.

So gehen Sie vor:

1. Programm abtippen und speichern.
2. Starten mit RUN
3. Anschalten des Checksummer 20 V3 mit SYS 955
4. Test: Geben Sie in einer freien Zeile ein: »1 REM« und drücken die RETURN-Taste. Am Bildschirm oben links sollten Sie die Prüfsumme <63> sehen.
5. Geben Sie ein Listing aus unserem Heft ein. Nach jeder Zeile wird die Zahl, die im Listing in Klammern < > steht, in den Bildschirm eingeblendet. Stimmen die Zahlen nicht überein, so liegt vermutlich ein Eingabefehler vor. **Die Zahl in den Klammern und auch die Klammern selbst, dürfen beim Abtippen nicht mit eingegeben werden!**
6. Dieser Checksummer 20 V3 bemerkt auch Vertauschungen von Zahlen und Buchstaben.
7. Abschaltung wird mit »SYS 58459« vollzogen.

Achtung: Nehmen Sie keine Kassetten-Operationen vor, wenn der Checksummer VC 20 eingeschaltet ist. Da das Betriebssystem den Kassettenpuffer mit Daten belegt, kann der Checksummer VC 20 überschrieben werden. Wollen Sie deshalb ein Programm auf (von) Kassette speichern (laden), so müssen Sie erst den Checksummer VC 20 abschalten.

Als Sicherung wird bei der Initialisierung geprüft, ob das zuletzt angesprochene Peripherie-Gerät der Kassettenrecorder war. Ist das der Fall, so werden die Betriebssystemroutinen LOAD und SAVE für die Benutzung gesperrt. Der Computer meldet bei Aufruf einer dieser beiden Routinen READY, ohne weitere Aktionen durchzuführen. Diese Sicherung kann man nach der Tipparbeit aufheben, wenn man den Checksummer VC 20 mit SYS 58459 abschaltet. Weiterhin wird dann durch gleichzeitiges Drücken der Tasten »Run-Stop & Restore« erreicht, daß die Betriebssystemroutinen LOAD und SAVE wieder eingerichtet werden.

– Bei Benutzung einer Diskettenstation brauchen Sie nicht darauf zu achten, daß bei LOAD beziehungsweise SAVE der Checksummer VC 20 überschrieben wird, da der Kassettenpuffer für die Diskettenstation normalerweise nicht genutzt wird. Deshalb können Sie die beiden Routinen weiterhin nor-

```
10 REM***************
11 REM*             *
12 REM* CHECKSUMMER *
13 REM*             *
14 REM* V3     VC20 *
15 REM*             *
16 REM*   WRITTEN   *
17 REM* MAERZ  1985 *
18 REM*      BY     *
19 REM*F. LONCZEWSKI*
20 REM***************
21 PRINT"{CLR,SPACE,RVSON}CHECKSUMMER V3 V
   C-20{RVOFF}"
22 PRINT"{2DOWN}EINEN MOMENT, BITTE..."
23 FOR I=827 TO 1019:READ A:POKE I,A
24 PS=PS+A+1:NEXT I
25 IF PS<>24464 THEN PRINT"{DOWN}PRUEFSUMM
   ENFEHLER !":END
26 SYS 981:PRINT"CHECKSUMMER AKTIVIERT."
27 PRINT"AN :SYS981"
28 PRINT"{DOWN}AUS:SYS58459, BEI CAS-{4SPA
   CE}SETTE ZUSAETZLICH{5SPACE}RUN/STOP &
   RESTORE"
29 PRINT"{DOWN}BEI AKTIVIERTEM CHECK-SUMME
   R KEIN";
30 PRINT" CASSETTEN-BETRIEB (LOAD, SAVE){2
   SPACE}ERLAUBT!":NEW
31 DATA 32,95,3,134,122,132,123,32,115,0,1
   70,240,243,162,255
32 DATA 134,58,144,10,162,0,134,255,32,121
   ,197,76,225,199,162
33 DATA 1,134,255,76,156,196,166,255,224,1
   ,240,3,76,96,197
34 DATA 160,2,169,0,170,133,254,177,95,240
   ,40,201,32,208,3
35 DATA 200,208,245,133,253,138,41,7,170,2
   [illegible],14,72,165,253,24
36 DATA 42,105,0,202,208,249,133,253,104,1
   70,232,165,253,24,101
37 DATA 254,133,254,76,119,3,192,4,48,219,
   198,214,165,214,72
38 DATA 162,3,169,32,157,1,4,189,209,3,32,
   210,255,202,16
39 DATA 242,166,254,169,0,32,205,221,169,6
   2,32,210,255,104,133
40 DATA 214,32,135,229,169,141,32,210,255,
   162,0,134,255,240,148
41 DATA 9,60,18,19,169,59,141,2,3,169,3,14
   1,3,3,165
42 DATA 186,201,1,208,16,169,116,141,48,3,
   141,50,3,169,196
43 DATA 141,49,3,141,51,3,173,136,2,141,17
   0,3,96

© 64'er
```

```
5 PRINT CHR$(14)                                   <242>
10 PRINT"{CLR}"                                    <254>
20 PRINT"Q****************************W"           <130>
30 PRINT"{4DOWN,2SPACE}TEST{SPACE,BLUE,6SP
   ACE}"                                           <022>
40 PRINT"BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB"           <108>

© 64'er
```

Bild 1. So könnte ein Teil eines Listings abgedruckt sein. In Zeile 10 müssen Sie nach den Anführungsstrichen die CLEAR/HOME-Taste drücken und nicht die Klammern mit dem Wort CLR. In Zeile 20 drücken Sie nach den Anführungsstrichen die Commodore-Taste und den Buchstaben Q, gefolgt von mehreren SHIFT- und Stern-Tasten und zum Schluß die Commodore-Taste und den Buchstaben W. In Zeile 30 ist es viermal die Cursor-nach-unten-Taste, gefolgt von zweimal die Leertaste, dann SHIFT und T und normal EST, zum Schluß noch einmal die Leertaste, die Farbtaste Blau (Control und 7) und sechsmal die Leertaste. Zeile 40 besteht lediglich aus mehreren Grafikzeichen, die mit der Commodore-Taste und B erzeugt werden.

```
5 PRINTCHR$(14)
10 PRINT"□"
20 PRINT"├──────────────────────────────┤"
30 PRINT"QQQQ  |EST ■      "
40 PRINT"▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚▚"
```

Bild 2. Auf dem Bildschirm oder Ihrem Drucker sieht das Listing (Bild 1) so aus.

mal nutzen, sofern der Computer bei der Initialisierung des Checksummer VC 20 feststellt, daß das zuletzt angesprochene Peripherie-Gerät nicht der Kassettenrecorder war.
- Wird eine Zeile gelöscht, also eine Zahl zwischen 0 und 65999 eingegeben und danach Return gedrückt, so wird eine Checksumme ausgegeben, die aber keine Bedeutung hat.

Sie können die Programme auch weiterhin ohne den Checksummer eintippen.

(F. Lonczewski/gk)

Hinweis: {13 SPACE} bedeutet 13mal die Leertaste drücken

Wie unsere Basic-Programme einzugeben sind

Unsere Basic-Listings enthalten keine Steuerzeichen mehr. Diese werden ersetzt durch Klartext und stehen zwischen geschweiften Klammern. Deshalb sind weder die Klammern noch was dazwischen steht, abzutippen, sondern die in Tabelle 1 aufgeführten Tasten zu drücken. Auf Ihrem Bildschirm erhalten Sie dann wieder die entsprechenden Grafikzeichen (siehe Bild 1 und 2).

Alle Grafikzeichen werden ebenfalls ersetzt durch unterstrichene oder überstrichene Großbuchstaben.

Unterstrichene Buchstaben bedeuten, daß Sie die SHIFT-Taste und den angegebenen Buchstaben drücken müssen, überstrichene jedoch die Commodore-Taste mit dem Buchstaben.

Auch hier erhalten Sie am Bildschirm das entsprechende Grafikzeichen und nicht etwa das im Listing erkennbare Zeichen (siehe Bild 1 und 2).

Die Leerzeichen zwischen den einzelnen Basic-Befehlen können beim Abtippen entfallen (ohne Einfluß auf die Checksumme zu nehmen). Dies ist besonders bei speicherkritischen Programmen wichtig.

Ebenso müssen Zeilen, die mehr als 80 Zeichen pro Zeile enthalten, mit den bekannten Abkürzungen für die Basic-Befehle (siehe auch das Handbuch zum C64, Anhang D, Seite 130) eingegeben werden.

CTRL steht für Control-Taste, so bedeutet [CTRL-A], daß Sie die Control-Taste und die Taste »A« drücken müssen. Im folgenden steht:

Klartext	Taste(n)
{DOWN}	Taste neben rechtem Shift, Cursor unten
{UP}	Shift-Taste & Taste neben rechtem Shift; Cursor hoch
{CLR}	Shift-Taste & 2. Taste ganz rechts oben
{INST}	Shift-Taste & Taste ganz rechts oben
{HOME}	2. Taste von ganz rechts oben
{DEL}	Taste ganz rechts oben
{RIGHT}	Taste ganz rechts unten
{LEFT}	Shift-Taste & Taste unten rechts
{SPACE}	Leertaste
{F1}	grauer Tastenblock rechts
{F3}	grauer Tastenblock rechts
{F5}	grauer Tastenblock rechts
{F7}	grauer Tastenblock rechts
{F2}	grauer Tastenblock rechts & Shift
{F4}	grauer Tastenblock rechts & Shift
{F6}	grauer Tastenblock rechts & Shift
{F8}	grauer Tastenblock rechts & Shift
{RETURN}	Shift-Taste & Return
{BLACK}	Control-Taste & 1
{WHITE}	Control-Taste & 2
{RED}	Control-Taste & 3
{CYAN}	Control-Taste & 4
{PURPLE}	Control-Taste & 5
{GREEN}	Control-Taste & 6
{BLUE}	Control-Taste & 7
{YELLOW}	Control-Taste & 8
{RVSON}	Control-Taste & 9
{RVOFF}	Control-Taste & 0
{ORANGE}	Commodore-Taste & 1
{BROWN}	Commodore-Taste & 2
{LIG.RED}	Commodore-Taste & 3
{GREY 1}	Commodore-Taste & 4
{GREY 2}	Commodore-Taste & 5
{LIG.GREEN}	Commodore-Taste & 6
{LIG.BLUE}	Commodore-Taste & 7
{GREY 3}	Commodore-Taste & 8

Tabelle 1. Die Steuerbefehle im Klartext für den C64/VC20/C16

Wenn Sie sich erst einmal an die in Klartext geschriebenen Steuerzeichen gewöhnt haben, werden Sie den Vorteil dieser Schreibweise erkennen. Der zu dem jeweiligen Steuerzeichen gehörende Klartext ist so verfaßt, daß Sie leicht die Taste beziehungsweise die Tastenkombination finden, die Sie drücken müssen.

Sparen mit dem VC20

Haben Sie vor, Ihr Geld gewinnbringend anzulegen? Mit diesem Programm können Sie die komplizierten Zins- und Zinseszinsberechnungen mit dem VC20 durchspielen, und sich so einen Überblick über die Gewinne verschaffen.

Um die Bedienung des Programms so einfach wie möglich zu gestalten, sind sämtliche Funktionen von Menüs aus erreichbar. Als besonderes »Bonbon« werden alle Zahlen formatiert dargestellt (Beispiel: Eingabe ».5«, Anzeige »0,50«).

Für den VC20 benötigt man mindestens eine 16 KByte Speichererweiterung.

»Zinsen« (siehe Listing) besteht aus acht Hauptprogrammen und den Unterprogrammen für Ein- und Ausgabeformatierung sowie einer Hardcopyroutine zur Ausgabe des Bildschirminhaltes auf Drucker.

Zur Eingabe sind folgende Tasten erlaubt:
- numerische Tasten und ».« für den Dezimalpunkt.
- »INST/DEL« für Korrekturen.
- »F1« Zurück zum Menü.
- »F7« Nochmal dieselbe Funktion ausführen.
- »F8« Hardcopy in Breitschrift für den Commodore-Drucker MPS 802. Will man Normalschrift verwenden, so muß in Zeile 60010 der Ausdruck »SI$=CHR$ (14)« durch »SI$=CHR$ (15)« ersetzt werden. In den Zeilen 60190 und 60200 kann der Druck-Tabulator ("20") nach Belieben geändert werden.

Bemerkungen zu den einzelnen Unterprogrammen:
Die grundlegende Zinsformel besteht immer aus den Faktoren Zinsen (in DM), Zinssatz (in Prozent), der Laufzeit (Jahre, Monate, Tage) und dem Grundkapital. Sie können nun auswählen, welchen der Faktoren Sie berechnet haben möchten:

Menü 1:
1. Kapital
2. Prozent (=Zinssatz)
3. Zinsen
4. Tage (=Laufzeit)
5. Zinseszinsen mit monatlich gleichbleibenden Einzahlungen

In diesem Unterprogramm werden einige Abkürzungen verwendet:

M.-Einz.	= monatliche Einzahlung
Prozent	= Prozentsatz
Jahre	= Vertragszeit einschließlich der Monate, die am Ende des Vertrages keine Einzahlungen erfordern (Ruhezeit, zum Beispiel beim 624 Mark-Gesetz).
Dav. keine Einz. M.	= Davon keine Einzahlungen im Monat zum Vertragsende (maximal 12 Monate).
Bonus auf Einz. Prozent	= Bonus auf Einzahlung in Prozent. Einige Banken geben am Ende der Vertragslaufzeit auf die gesamten Einzahlungen einen Bonus.

Beispiel zu Menüpunkt 5 (Zinseszinsen mit monatlich gleichbleibenden Einzahlungen):

52 Mark monatliche Einzahlung
Darauf 3 Prozent Zinsen
Laufzeit: 7 Jahre
Davon das letzte Jahr keine Einzahlung (Ruhezeit)
14 Prozent Bonus auf die gesamten Einzahlungen, wenn der Vertrag eingehalten wird.

Die Eingabe sieht wie folgt aus:

M.-Einz.	>52,	Anzeige:	>52.00<
Prozent	> 3,	Anzeige:	> 3.00<
Jahre	> 7,	Anzeige:	> 7 <
Dav. keine Einz. M.	>12,	Anzeige:	>12 <
Bonus auf Einz. %	>14,	Anzeige:	>14.00<

Ergebnis:		
	Eigenleistung	>3744,00<
	Zinseszinsen	> 480,93<
	Bonus	> 524,16<
	Gesamt	>4749,09<

Menü 2:

1. Errechnen Zinstage

Dieser Menüpunkt dient zur Errechnung der Anzahl an Zinstagen zwischen zwei Daten.

Beispiel:

Wieviele Zinstage sind es vom 25.2. bis zum 5.10. (des gleichen Jahres)?

Eingabe: >2502< und >0510<

Ergebnis: 7 Monate und 10 Tage

2. Was kostet ein Kredit

Hier können Sie sich informieren, wie stark die Aufnahme eines Kredits Ihren Geldbeutel belasten wird. Es werden folgende Abkürzungen verwendet:

Kred.-Sum.	= Kreditsumme. Die Anzahlung muß vorher abgezogen werden.
Bearbeitg.-Gebühr in %	= Einmalige Bearbeitungsgebühr in Prozent. Sind keine Gebühren zu entrichten, nur die Returntaste drücken.
Laufzeit/Monat	= Laufzeit in Monaten.
% in Monat	= Prozentsatz pro Monat.

Zu Menü 2, »Was kostet ein Kredit«:

Aus Platzgründen werden nur die tatsächlichen und nicht die effektiven Zinsen berechnet.

Die tatsächlichen Zinsen werden aus »% in Monaten«, mit 12 multipliziert, plus der »Bearbeitg.-Gebühr in %« auf ein Jahr errechnet. Der effektive Zinssatz liegt weitaus höher. Man müßte die echten Zinsen errechnen und im Dreisatz den tatsächlichen Zinsen entgegenhalten. Jeder Kreditgeber hat da so seine eigene Philosophie! Interessant sind natürlich für den Einzelnen die Kreditkosten, die bezahlt werden müssen.

Für eventuelle Konkursanmeldungen oder Bankrotte wird seitens des Autors (und der 64'er Redaktion) keinerlei Haftung übernommen!

(Helmut Gurtner/tr)

```
1 REM************************            <142>
2 REM* GURTNER HELMUT         *           <082>
3 REM* GOETHESTR. 9           *           <120>
4 REM* 5120 HERZOGENRATH      *           <250>
5 REM* TEL: 02406/64433       *           <060>
8 REM************************            <149>
10 POKE 36879,25:PRINT"{CLR}"             <219>
20 A$="KAPITAL"                           <096>
30 A1$="PROZENT"                          <109>
40 A2$="ZINSEN"                           <143>
50 A3$="JAHRE"                            <123>
60 A4$="MONATE"                           <210>
70 A5$="TAGE"                             <051>
80 A6$="ZINSESZ."                         <203>
90 A7$="GESAMT"                           <190>
100 A8$="{RVSON}ZURUECK MENUE (F1){4SPACE,
    RVOFF}"                               <222>
110 A9$="{UP,RVSON}NOCHMAL (F7) H.-C.(F8){
    RVOFF}"                               <158>
120 B$="{HOME,18DOWN}"                    <174>
130 B1$="**********************"          <042>
140 B2$="J.-EINZ."                        <029>
150 B3$="M.-EINZ."                        <176>
200 PRINT"{CLR,BLACK,6SPACE}M E N U E{2SPA
    CE}1":PRINT                           <228>
210 PRINT"{RED}ZINSBERECHNUNGEN{BLUE,6SPAC
    E}"B1$                                <098>
220 PRINT"1{2SPACE}"A$:PRINT              <121>
230 PRINT"2{2SPACE}"A1$:PRINT             <170>
240 PRINT"3{2SPACE}"A2$:PRINT             <069>
250 PRINT"4{2SPACE}"A5$:PRINT             <224>
260 PRINT"5{2SPACE}ZINSESZINSEN MIT"      <052>
270 PRINT"{3SPACE}MONATL. EINZAHLUNG":PRIN
    T                                     <145>
280 PRINT"6{2SPACE}ZINSESZINSEN MIT"      <088>
290 PRINT"{3SPACE}EINMALIGER EINZ.":PRINT <189>
300 PRINT"7{2SPACE}WEIT. MENUEAUSWAHL"    <124>
310 GET W$:IF W$=""THEN 310               <244>
320 IF ASC(W$)<49 OR ASC(W$)>55 THEN 310  <216>
330 W=VAL(W$):ON W GOTO 600,1000,1400,1800
    ,2200,2600,340                        <082>
340 PRINT"{CLR,BLACK,6SPACE}M E N U E{2SPA
    CE}2":PRINT                           <122>
341 PRINT"{RED}ZINSBERECHNUNGEN{BLUE,6SPAC
    E}"B1$                                <231>
350 PRINT"1{2SPACE}ERRECHNEN ZINSTAGE":PRI
    NT                                    <173>
360 PRINT"2{2SPACE}WAS KOSTET EIN"        <036>
370 PRINT"{3SPACE}KREDIT":PRINT           <063>
390 PRINT"3{2SPACE}ZURUECK MENUE":PRINT   <055>
420 GET W1$:IF W1$=""THEN 420             <169>
430 IF ASC(W1$)<49 OR ASC(W1$)>55 THEN 420 <215>
450 W1=VAL(W1$):ON W1 GOTO 3000,7000,200  <183>
460 GOTO 200                              <142>
600 L=10:REM **** KAPITAL ****            <100>
610 PRINT"{CLR,4SPACE}Z * 100 * 360"      <069>
620 PRINT"K = **************"             <037>
630 PRINT"{7SPACE}P * T":PRINT            <206>
640 PRINT A2$:PRINT A1$:PRINT A3$:PRINT A4
    $:PRINT A5$:PRINT B1$:PRINT A$        <134>
720 Z3=9:Z2=5:GOSUB 3520:Z2=6:GOSUB 3510:G
    OSUB 3530:GOSUB 5000                  <166>
725 IF Z=0 OR P=0 OR T=O THEN GOSUB 6200:G
    OTO 600                               <027>
730 K=(Z*100*360)/(P*T)                   <066>
740 X=K:GOSUB 4500:Z2=12:PRINT"{BLACK}":GO
    SUB 4950:PRINT"{BLUE}":GOSUB 6100     <243>
745 PRINT B$:PRINT A8$:PRINT A9$          <108>
750 GET W1$:IF W1$=""THEN 750             <121>
760 IF W1$=CHR$(133)THEN GOTO 200         <138>
770 IF W1$=CHR$(136)THEN GOTO 600         <176>
780 IF W1$=CHR$(140)THEN GOSUB 60000      <208>
790 GOTO 750                              <084>
1000 L=10:REM **** PROZENT ****           <220>
1010 PRINT"{CLR,4SPACE}Z * 100 * 360"     <215>
1020 PRINT"P = **************"            <008>
1030 PRINT"{7SPACE}K * T":PRINT           <017>
1040 PRINT A$:PRINT A2$:PRINT A3$:PRINT A4
     $:PRINT A5$:PRINT B1$:PRINT A1$      <229>
1060 Z3=9:Z2=5:GOSUB 3500:Z2=6:GOSUB 3520:
     GOSUB 3530:GOSUB 5000                <001>
1065 IF Z=0 OR K=0 OR T=0 THEN GOSUB 6200:
     GOTO 1000                            <044>
1070 P=(Z*100*360)/(K*T)                  <164>
```

Listing zu »Zinsen VC20«

```
1080 X=P:GOSUB 4500:Z2=12:PRINT"{BLACK}":G
     OSUB 4950:PRINT"{BLUE}":GOSUB 6100      <156>
1085 PRINT B$:PRINT A8$:PRINT A9$             <196>
1090 GET W1$:IF W1$=""THEN 1090               <155>
1100 IF W1$=CHR$(133)THEN GOTO 200            <226>
1110 IF W1$=CHR$(136)THEN GOTO 1000           <067>
1120 IF W1$=CHR$(140)THEN GOSUB 60000         <038>
1130 GOTO 1090                                <087>
1400 L=10:REM **** ZINSEN                     <024>
1410 PRINT"{CLR,4SPACE}K * P * T":Z3=9        <213>
1420 PRINT"Z = ***************"               <060>
1430 PRINT"{5SPACE}100 * 360":PRINT           <151>
1440 PRINT A$:PRINT A1$:PRINT A3$:PRINT A4
     $:PRINT A5$:PRINT B1$                    <233>
1450 PRINT A2$:PRINT A$:PRINT:PRINT A7$       <071>
1460 Z2=5:GOSUB 3500:Z2=6:GOSUB 3510:GOSUB
      3530:GOSUB 5000                         <160>
1465 IF K=0 OR P=0 OR T=0 THEN GOSUB 6200:
     GOTO 1400                                <203>
1470 Z=(K*P*T)/(100*360)                      <050>
1480 X=Z:GOSUB 4500:Z2=12:PRINT"{BLACK}":G
     OSUB 4950:PRINT"{BLUE}"                  <078>
1490 X=K:GOSUB 4500:Z2=13:GOSUB 4950          <098>
1500 KZ=K+Z:X=KZ:GOSUB 4500:Z2=15:GOSUB 49
     50:GOSUB 6100                            <063>
1505 PRINT B$:PRINT A8$:PRINT A9$             <106>
1510 GET W1$:IF W1$=""THEN 1510               <054>
1520 IF W1$=CHR$(133)THEN GOTO 200            <136>
1530 IF W1$=CHR$(136)THEN GOTO 1400           <237>
1540 IF W1$=CHR$(140)THEN GOSUB 60000         <206>
1550 GOTO 1510                                <078>
1800 L=10:REM **** TAGE ****                  <024>
1810 PRINT"{CLR,4SPACE}Z * 100 * 360"         <255>
1820 PRINT"T = ***************"               <135>
1830 PRINT"{7SPACE}K * P":PRINT:Z3=9:GOSUB
      6100                                    <037>
1840 PRINT A$:PRINT A1$:PRINT A2$:PRINT B1
     $                                        <236>
1850 PRINT A5$:PRINT B1$:PRINT"{DOWN}"A3$:
     PRINT"{DOWN}"A4$:PRINT"{DOWN}"A5$        <209>
1870 Z2=5:GOSUB 3500:Z2=6:GOSUB 3510:Z2=7:
     GOSUB 3520                               <212>
1875 IF K=0 OR P=0 OR Z=0 THEN GOSUB 6200:
     GOTO 1800                                <155>
1880 T=(Z*100*360)/(K*P)                      <196>
1890 X=T:GOSUB 4500:Z2=10:PRINT"{BLACK}":G
     OSUB 4950:PRINT"{BLUE}":T=INT(T+.5)      <234>
2000 IF T<360 THEN 2020                       <083>
2010 T=T-360:J=J+1:IF T>=360 THEN 2010        <112>
2020 IF T<30 THEN 2040                        <037>
2030 T=T-30:M=M+1:IF T>=30 THEN 2030          <243>
2040 PRINT LEFT$(B$,14);:PRINT TAB(14)J:PR
     INT:PRINT TAB(14)M:PRINT:PRINT TAB(14
     )T                                       <216>
2050 PRINT"{BLUE}":PRINT B$:PRINT A8$:PRIN
     T A9$                                    <183>
2060 GET W1$:IF W1$=""THEN 2060               <232>
2070 IF W1$=CHR$(133)THEN GOTO 200            <180>
2080 IF W1$=CHR$(136)THEN GOTO 1800           <253>
2090 IF W1$=CHR$(140)THEN GOSUB 60000         <248>
2100 GOTO 2060                                <208>
2200 PRINT CHR$(147)"{BLACK}ZINSESZINSEN M
     IT":L=10:Z3=9                            <078>
2210 PRINT"GLEICHBLEIBENDER"                  <167>
2220 PRINT"MONATL. EINZAHLUNG{BLUE}":PRINT
     "{DOWN}"B3$:PRINT A1$:PRINT A3$          <236>
2225 PRINT"DAV. KEINE":PRINT"EINZ. M.":PRI
     NT"BONUS AUF"                            <154>
2226 PRINT"EINZ. %":PRINT B1$:PRINT"EIGENL
     ."                                       <201>
2230 PRINT A6$:PRINT"BONUS":PRINT:PRINT A7
     $                                        <164>
2240 Z2=5:Z3=11:GOSUB 3500:Z2=6:GOSUB 3510
     :Z2=7:GOSUB 3525                         <030>
2242 Z2=9:GOSUB 4000:KE=N:GOSUB 4800:GOSUB
      4950:IF KE>12 THEN 2242                 <048>
2244 Z2=11:GOSUB 4000:BP=N:GOSUB 4500:GOSU
     B 4950                                   <182>
2250 KG=0:KK=0:KZ=0:MZ=12:Y=12:KB=0:BD=0:Z
     G=0                                      <116>
2260 M=J*12:M1=M-KE:M2=M1:IF M1<12 THEN MZ
     =M1                                      <097>
2265 IF M=KE THEN PRINT"{CLR,2DOWN}FALSCHE
      ANGABEN":FOR I=1 TO 2000:NEXT:GOTO 2
     200                                      <139>
2270 GOSUB 2500                               <026>
2280 M2=M2-12                                 <136>
2290 IF M2=>12 THEN GOTO 2510                 <212>
2295 KB=KB+KX                                 <186>
2300 IF KE=12 THEN ZZ=KB*P/100:KB=KB+ZZ       <128>
2306 IF M2<12 AND M2>0 THEN MZ=M2:GOSUB 25
     00:ZZ=KB*P/100:KB=KB+KX+ZZ               <067>
2310 KK=K*M1:BD=KK*BP/100:ZG=KB-KK:KG=KB+B
     D                                        <154>
2320 X=KK:GOSUB 4500:Z2=14:Z3=11:GOSUB 495
     0                                        <066>
2330 X=ZG:GOSUB 4500:Z2=15:GOSUB 4950         <115>
2340 X=BD:GOSUB 4500:Z2=16:GOSUB 4950         <158>
2350 X=KG:GOSUB 4500:Z2=18:GOSUB 4950:GOSU
     B 6100                                   <091>
2355 PRINT B$:PRINT A8$:PRINT A9$             <196>
2360 GET W1$:IF W1$=""THEN 2360               <025>
2370 IF W1$=CHR$(133)THEN GOTO 200            <226>
2380 IF W1$=CHR$(136)THEN GOTO 2200           <198>
2390 IF W1$=CHR$(140)THEN GOSUB 60000         <038>
2400 GOTO 2360                                <046>
2500 KA=0:ZA=0:KX=0:Y=12                      <021>
2501 FOR I=1 TO MZ                            <161>
2502 ZZ=(K*P*Y)/(100*12)                      <117>
2503 ZZ=INT(ZZ*10000+.5)/10000                <164>
2504 ZA=ZA+ZZ:KA=KA+K:Y=Y-1                   <094>
2505 NEXT I                                   <047>
2506 KX=KA+ZA:RETURN                          <060>
2510 KB=KB+KX                                 <147>
2512 ZB=KB*P/100                              <202>
2514 ZB=INT(ZB*10000+.5)/10000                <215>
2516 KB=KB+ZB                                 <124>
2518 GOTO 2280                                <213>
2600 PRINT CHR$(147)"{BLACK}ZINSESZINSEN B
     EI EIN-":L=10:Z3=9:KZ=0:ZG=0             <036>
2610 PRINT"MALIGER EINZAHLUNG{BLUE}":PRINT
     :PRINT                                   <093>
2620 PRINT A$:PRINT A1$:PRINT A3$:PRINT B1
     $:PRINT A$:PRINT A6$:PRINT:PRINT A7$     <157>
2640 Z2=5:GOSUB 3500:Z2=6:GOSUB 3510:Z2=7:
     GOSUB 3525                               <230>
2650 ZZ=0:KZ=K                                <249>
2660 FOR I=1 TO J                             <103>
2670 ZZ=KZ*P/100                              <048>
2680 ZZ=INT(ZZ*10000+.5)/10000                <087>
2690 KZ=KZ+ZZ                                 <034>
2700 NEXT I                                   <244>
2710 ZG=KZ-K                                  <151>
2720 X=K:GOSUB 4500:Z2=10:GOSUB 4950          <055>
2730 X=ZG:GOSUB 4500:Z2=11:PRINT"{BLACK}":
     GOSUB 4950:PRINT"{BLUE}"                 <013>
2740 X=KZ:GOSUB 4500:Z2=13:GOSUB 4950         <125>
2745 PRINT B$:PRINT A8$:PRINT A9$             <076>
2750 GET W1$:IF W1$=""THEN 2750               <163>
2760 IF W1$=CHR$(133)THEN GOTO 200            <106>
2770 IF W1$=CHR$(136)THEN GOTO 2600           <082>
2780 IF W1$=CHR$(140)THEN GOSUB 60000         <174>
2790 GOTO 2750                                <214>
3000 PRINT"{CLR,RED}ERRECHNEN ZINSTAGE":PR
     INT:L=8:Z2=5:Z3=8:J=0:M=0:T=0            <045>
3010 PRINT"{8SPACE,BLACK}TTMMJJJJ{BLUE}":P
     RINT                                     <190>
3020 PRINT"ANFANG:":PRINT                     <018>
3030 PRINT"ENDE:":PRINT:PRINT                 <059>
3040 PRINT"I S T":PRINT                       <254>
3050 PRINT B1$                                <015>
3060 PRINT"ODER JAHR/E":PRINT                 <206>
3070 PRINT"{5SPACE}MONAT/E":PRINT             <041>
3080 PRINT"{5SPACE}TAG/E"                     <245>
3100 GOSUB 4000:AA$=NN$:GOSUB 6000:Z2=7:GO
     SUB 4000:E$=NN$:GOSUB 6000               <005>
3110 JA=VAL(MID$(AA$,5,4)):MA=VAL(MID$(AA$
     ,3,2)):TA=VAL(MID$(AA$,1,2)):GOSUB 60
     00                                       <074>
3130 JE=VAL(MID$(E$,5,4)):ME=VAL(MID$(E$,3
     ,2)):TE=VAL(MID$(E$,1,2)):GOSUB 6000     <058>
3140 J=JE-JA                                  <182>
3150 T=(J*360)+(ME*30)+TE-(MA*30)-TA:J=0      <127>
3160 PRINT LEFT$(B$,10);:PRINT TAB(7)T;:PR
     INT A5$                                  <026>
3170 IF T<360 THEN 3190                       <045>
3180 T=T-360:J=J+1:IF T>=360 THEN 3180        <043>
3190 IF T<30 THEN 3210                        <186>
```

Listing zu »Zinsen VC 20«. Beachten Sie unbedingt, daß ihr VC 20 eine 16-KByte-Speichererweiterung besitzen muß.

```
3200 T=T-30:M=M+1:IF T>=30 THEN 3200            <205>
3210 X=J:Z2=14:Z3=11:PRINT"{BLACK}":GOSUB
     4800:GOSUB 4950:X=M:Z2=16:GOSUB 4800:
     GOSUB 4950                                 <223>
3220 X=T:Z2=18:GOSUB 4800:GOSUB 4950:PRINT
     "{BLUE}":GOSUB 6100                        <118>
3225 PRINT B$:PRINT A8$:PRINT A9$               <048>
3230 GET W1$:IF W1$=""THEN 3230                 <254>
3240 IF W1$=CHR$(133)THEN 340                   <089>
3250 IF W1$=CHR$(136)THEN 3000                  <217>
3260 IF W1$=CHR$(140)THEN GOSUB 60000           <146>
3270 GOTO 3230                                  <050>
3500 GOSUB 4000:K=N:GOSUB 4500:GOSUB 4950:
     RETURN                                     <183>
3510 GOSUB 4000:P=N:GOSUB 4500:GOSUB 4950:
     RETURN                                     <198>
3520 GOSUB 4000:Z=N:GOSUB 4500:GOSUB 4950:
     RETURN                                     <218>
3525 GOSUB 4000:J=N:GOSUB 4800:GOSUB 4950:
     RETURN                                     <144>
3530 Z2=7:GOSUB 4000:J=N:GOSUB 4800:GOSUB
     4950                                       <021>
3540 Z2=8:GOSUB 4000:M=N:GOSUB 4800:GOSUB
     4950                                       <160>
3550 Z2=9:GOSUB 4000:T=N:GOSUB 4800:GOSUB
     4950:RETURN                                <031>
4000 REM  GET-SCHLEIFE                          <227>
4030 PRINT LEFT$(B$,Z2);:PRINT TAB(Z3)          <028>
4040 Z1=0:NN$="":PRINT"←";CHR$(157);            <011>
4050 GET N$:IF N$=""THEN 4050                   <114>
4055 IF N$=CHR$(140)THEN GOSUB 60000            <244>
4060 IF N$=CHR$(20)AND Z1>0 THEN 4130           <168>
4070 IF N$=CHR$(13)THEN 4140                    <071>
4080 IF N$<CHR$(46)OR N$>CHR$(57)THEN 4050      <120>
4090 PRINT N$;"←";CHR$(157);                    <029>
4100 NN$=NN$+N$                                 <244>
4110 Z1=Z1+1:IF Z1=L THEN 4140                  <018>
4120 GOTO 4050                                  <180>
4130 Z1=Z1-1:NN$=LEFT$(NN$,Z1):PRINT CHR$(
     157);"← ";CHR$(157);CHR$(157);:GOTO 4
     050                                        <227>
4140 PRINT" ";:N=VAL(NN$):X=N:RETURN            <206>
4500 REM****************                        <070>
4510 REM FORMATIERUNG                           <221>
4520 REM**************                          <006>
4600 X=INT(X*100+.5)/100                        <123>
4610 X$=" "+STR$(X)                             <178>
4615 IF MID$(X$,LEN(X$)-2,1)="."THEN 4640       <126>
4620 IF MID$(X$,LEN(X$)-1,1)<>"."THEN X$=X
     $+".00"                                    <248>
4630 IF MID$(X$,LEN(X$)-1,1)="."THEN X$=X$
     +"0"                                       <023>
4640 IF MID$(X$,LEN(X$)-3,1)=" "THEN X$="{
     2SPACE}0"+RIGHT$(X$,3)                     <106>
4650 IF MID$(X$,LEN(X$)-3,1)="-"THEN X$="
     -0"+RIGHT$(X$,3)                           <245>
4660 X$=RIGHT$("{7SPACE}"+X$,10)                <218>
4670 RETURN                                     <156>
4800 X$=STR$(X)                                 <136>
4810 X$="{7SPACE}"+X$+"{3SPACE}"                <057>
4820 X$=RIGHT$(X$,10)                           <103>
4830 RETURN                                     <060>
4950 PRINT LEFT$(B$,Z2);:PRINT TAB(Z3)X$:G
     OSUB 6000:RETURN                           <035>
5000 REM ERRECHNEN TAGE                         <194>
5010 T=T+(J*360)+(M*30)                         <132>
5020 RETURN                                     <252>
6000 NN$="":X=0:N$=""                           <227>
6010 RETURN                                     <226>
6100 K=0:Z=0:P=0:J=0:M=0:T=0:RETURN             <176>
6200 PRINT"{CLR,2DOWN}FEHLENDE ODER FALSCH
     E":PRINT:PRINT"EINGABEN":FOR I=1 TO 2
     500:NEXT:RETURN                            <225>
7000 PRINT"{CLR,BLACK,4SPACE}KREDIT - KOST
     EN{4SPACE,BLUE}":L=10:Z1=0:Z2=0:KK=0:
     EV=0:MZ=0                                  <230>
7010 REM PRINT"KREDIT{BLUE}":PRINT              <122>
7020 PRINT"KRED.-SUM.":PRINT                    <224>
7030 PRINT"BEARBEITG.-"                         <144>
7040 PRINT"GEBUEHR IN %":PRINT                  <177>
7050 PRINT"LAUFZ./MONAT":PRINT                  <170>
7055 PRINT"% IN MON.":PRINT                     <122>
7060 PRINT"{BLACK}KREDITKOSTEN":PRINT           <197>
7070 PRINT"TATSAECHL. ":PRINT"ZINSEN IN %"
     :PRINT                                     <116>
7080 PRINT"MONATL. ZU":PRINT"ZAHLEN{BLUE}"      <123>
7100 Z2=3:Z3=12:GOSUB 4000:KS=N:GOSUB 4500
     :GOSUB 4950                                <031>
7110 Z2=6:Z3=16:GOSUB 4000:BZ=N:GOSUB 4500
     :Z3=12:GOSUB 4950                          <067>
7120 Z2=8:Z3=16:GOSUB 4000:LZ=N:GOSUB 4800
     :Z3=12:GOSUB 4950                          <234>
7130 Z2=10:Z3=16:GOSUB 4000:PM=N:GOSUB 450
     0:Z3=12:GOSUB 4950                         <113>
7135 IF KS=0 OR LZ=0 OR PM=0 THEN GOSUB 62
     00:GOTO 7000                               <170>
7220 Z1=PM*LZ:REM % * LAUFZ.                    <165>
7230 Z1=(KS*Z1)/100:REM ZINSEN-LAUFZEIT         <226>
7240 Z2=(KS*BZ)/100:REM ZINSEN-BEARB.           <197>
7250 KK=Z1+Z2:REM KREDITKOSTEN                  <138>
7260 EV=(KK*100*12)/(KS*LZ):REM EFF.-ZINS.      <134>
7270 MZ=(KS+KK)/LZ:REM M. ZU ZAHLEN             <242>
7280 X=KK:Z2=12:Z3=12:PRINT"{BLACK}":GOSUB
     4500:PRINT"{BLACK}":GOSUB 4950             <153>
7290 X=EV:Z2=15:GOSUB 4500:GOSUB 4950           <051>
7300 X=MZ:Z2=18:GOSUB 4500:GOSUB 4950:PRIN
     T"{BLUE}"                                  <101>
7305 PRINT B$:PRINT A8$:PRINT A9$               <064>
7330 GET W1$:IF W1$=""THEN 7330                 <037>
7340 IF W1$=CHR$(133)THEN 340                   <125>
7350 IF W1$=CHR$(136)THEN 7000                  <254>
7360 IF W1$=CHR$(140)THEN GOSUB 60000           <182>
7370 GOTO 7330                                  <134>
60000 REM * HARD-COPY *                         <114>
60010 SI$=CHR$(14):BS$=CHR$(8):PO$=CHR$(16
      )                                         <053>
60020 QT$=CHR$(34)                              <012>
60030 MF$=CHR$(145):VR=PEEK(648)*256            <127>
60040 OPEN 4,4:PRINT#4                          <210>
60050 FOR CL=0 TO Z2:QF=0:AS$=MF$:FOR RO=0
       TO 21                                    <146>
60060 SC=PEEK(VR+22*CL+RO)                      <190>
60070 IF SC=34 THEN QF=1-QF                     <169>
60080 IF SC<>162 THEN 60110                     <113>
60090 QF=1-QF:IF QF=1 THEN AS$=AS$+QT$:GOT
      O 60170                                   <015>
60100 AS$=AS$+QT$:GOTO 60170:GOTO 60130         <238>
60110 IF QF=1 AND(SC>=128)THEN SC=SC-128:G
      OTO 60130                                 <141>
60120 IF SC>=120 THEN SC=SC-128:RF=1            <138>
60130 IF SC<32 OR SC>95 THEN AS=SC+64:GOTO
       60160                                    <032>
60140 IF SC>31 AND SC<64 THEN AS=SC:GOTO 6
      0160                                      <189>
60150 IF SC>63 AND SC<96 THEN AS=SC+32:GOT
      O 60160                                   <131>
60160 AS$=AS$+CHR$(AS)                          <168>
60170 IF RF=1 THEN RF=0                         <163>
60180 NEXT RO                                   <123>
60190 IF QF=0 THEN PRINT#4,SI$PO$"20"AS$:G
      OTO 60210                                 <063>
60200 PRINT#4,SI$;PO$;"20";AS$;QT$              <232>
60210 NEXT CL:CLOSE 4:RETURN                    <209>
```

Listing zu »Zinsen VC 20« (Schluß)



Datei-verwaltung für den C 16

Ein leistungsfähiges Programm zur Verwaltung beliebiger Daten gehört zu jedem Computer. »Datamaster« ist speziell auf den C16/116 mit Diskettenlaufwerk zugeschnitten. Der geringe Speicherplatz bereitet dabei keine Probleme.

Ein Computer wird für viele Zwecke eingesetzt. Je billiger er ist, desto mehr wird anscheinend mit ihm gespielt. Der C 16 ist jedoch ein Computer, der sehr leistungsfähig ist. Sein Basic ist einfach hervorragend - wesentlich komfortabler als zum Beispiel das des C 64. Sein einziges Handicap ist lediglich der geringe Speicherplatz. Das ist vor allem bei Programmen relevant, die eine Menge Daten zur Verwaltung haben. Doch es gibt Möglichkeiten, viele Daten auch mit geringem RAM-Speicherplatz zu verarbeiten. »Datamaster« nutzt diese konsequent aus. Allerdings ist dazu ein Disketten-Laufwerk notwendig. *Mit einer Datasette ist das Programm nicht lauffähig.* Doch nun zum Programm selbst.

Vor dem Start

Beachten Sie bei der Arbeit mit dem Programm bitte unbedingt die folgenden grundlegenden Punkte:

- Verlassen Sie das Programm niemals durch Ausschalten des Computers. Wählen Sie immer den dafür vorgesehenen Programmpunkt »Verlassen des Programms« an. Sollten Sie das Programm auf nicht vorgesehene Weise verlassen, kann es zum Verlust Ihrer Daten kommen.
- Wechseln Sie niemals eine eingelegte Diskette, wenn Sie das Programm nicht dazu aufgefordert hat. Es kann zum Verlust Ihrer Daten kommen.
- Wenn Sie eine neue Datei aufbauen, muß mit Hilfe des Programmpunktes »Neue Datei aufbauen« eine »Eingabemaske« erstellt werden. Das erste Feld dieser Eingabemaske ist immer ein »Index-Feld«. Dieses Index-Feld erlaubt eine besonders schnelle Suche nach Datensätzen. Wählen Sie daher als erstes Feld Ihrer Maske bitte jenes aus, über das Sie am häufigsten suchen (zum Beispiel »Name« in einer Adreß-Datei). Sie können bei der schnellen Suche über diesen Index noch weitere Suchkriterien in andere Felder der Maske eingeben (zum Beispiel »Name: Müller«, »Ort: Mannheim«). In diesem Fall werden nur »Müller« ausgegeben, die in »Mannheim« wohnen. Die Suche verläuft weiterhin schnell. Sollten Sie jedoch kein Suchkriterium in das erste Feld, das Index-Feld, eingeben, müssen Sie mit einer erheblich langsameren Suche nach Datensätzen rechnen als mit Kriterium.

Nötige Hardware
- ein Computer Commodore C 16 oder C 116
- Floppy-Laufwerk VC-1541
- Monitor oder Fernsehgerät
- eventuell Drucker

Zum Aufbau einer Datei benötigt der Datamaster eine leere und bereits formatierte Diskette. Alle eventuell auf dieser Diskette vorhandenen Informationen werden beim Aufbauen einer Datei unwiederbringlich gelöscht. Verwenden Sie daher niemals Ihre Programmdiskette zum Aufbau einer Datei und schützen Sie diese mit einem Schreibschutz.

Eigenschaften von Datamaster und Datamaster-Dateien
- Eine Datei kann maximal 662 Datensätze aufnehmen.
- Ein Datensatz kann eine maximale Länge von 245 Zeichen besitzen.
- Ein Datensatz (zum Beispiel eine Adresse) kann aus maximal 10 einzelnen Feldern bestehen (zum Beispiel »Name«, »Adresse«, »Telefonnummer«...).
- Das erste Feld einer jeden Datei ist ein »Index-Feld«. Ein solches Index-Feld erlaubt eine besonders schnelle Suche nach Daten.

Laden und Starten des Programms

Drücken Sie bitte die Funktionstaste F2 (DLOAD"), geben Sie ein: `DATAMASTER` und drücken Sie »RETURN«. Nachdem das Programm geladen wurde, geben Sie ein: `RUN` und drücken Sie wiederum »RETURN«.

Wenn Sie den Datamaster zum ersten Mal benutzen, müssen Sie zuerst eine Datei aufbauen. Wählen Sie daher bitte den Programmpunkt »Neue Datei aufbauen« an. Zum Aufbau einer neuen Datei wird eine leere, jedoch bereits formatierte Diskette benötigt.

Wenn Sie mit einer bereits aufgebauten Datei arbeiten wollen, wählen Sie bitte den Programmpunkt »Datendiskette wechseln« an. Sie werden nun aufgefordert, eine Datendiskette einzulegen. Legen Sie die Diskette ein, auf der sich die von Ihnen gewünschte Datei befindet und drücken Sie eine beliebige Taste. Sie können nun die Arbeit mit dieser Datei aufnehmen.

Anwählen der einzelnen Programmfunktionen

Nach dem Starten des Programms sehen Sie auf dem Bildschirm »Auswahlmenü«.

Um die von Ihnen gewünschte Funktion anzuwählen, drücken Sie bitte die Taste mit dem entsprechenden Anfangsbuchstaben der Funktion, zum Beispiel »E« für »EINGABE VON DATENSÄTZEN« oder »N« für: »NEUE DATEI AUFBAUEN«.

Beenden der Arbeit mit dem Datamaster

Wenn Sie die Arbeit mit dem Programm beenden wollen, wählen Sie bitte die Funktion »VERLASSEN DES PROGRAMMS« an.

Beenden Sie die Arbeit nie auf andere Art und Weise, zum Beispiel durch Ausschalten des Computers.

Die Eingabemaske

Bei den einzelnen Programmfunktionen müssen die Daten, die das Programm zur Durchführung der Funktion von Ihnen benötigt, in eine »Eingabemaske« eingegeben werden.

Zur Eingabe der Daten in diese Maske besitzen Sie folgende Bewegungsmöglichkeiten:

- Mit Hilfe der Tasten »Cursor rechts« beziehungsweise »Cursor links« können Sie sich innerhalb eines Feldes der Maske hin und her bewegen.
- Mit Hilfe der Taste »Delete« können Sie ein eingegebenes Zeichen löschen, außer wenn Sie sich in der Funktion »NEUE DATEI AUFBAUEN« befinden.
- In ein tiefergelegenes Feld kommen Sie mit Hilfe der Taste »Cursor nach unten«, in ein Höhergelegenes durch die Taste »Cursor nach oben«.

Um die gesamte Dateneingabe in die Maske abzuschließen, drücken Sie bitte die Funktionstaste F1.

Sie können jede Programmfunktion vorzeitig durch Drücken der Funktionstaste HELP verlassen. Der jeweilige Programmteil wird dann ohne Durchführung der Funktion verlassen und Sie kehren zum Auswahlmenü zurück.

Die einzelnen Programmfunktionen

Neue Datei aufbauen

Dieses ist die erste Funktion, die Sie benötigen. Wenn Sie das Programm zum ersten Mal geladen und gestartet haben, wählen Sie diese Funktion an. Halten Sie bitte eine leere und formatierte Diskette bereit.

Sie können nun Ihre erste Datei aufbauen und auch später durch Anwahl dieser Funktion weitere von Ihnen benötigte Dateien. Für jede Datei wird eine eigene Diskette benötigt.

Nach Anwahl dieser Funktion sehen Sie auf dem Bildschirm die bereits besprochene Eingabemaske. In diese können Sie nun die Form Ihrer individuellen Datensätze eingeben. Die Bezeichnung der einzelnen Felder ihrer Datensätze, die Länge dieser Felder (maximal 67 Zeichen) und den Typ (alphanumerisch, das heißt sowohl Buchstaben als auch Zahlen als auch Sonderzeichen werden zugelassen, oder numerisch, das heißt nur Zahlen und der Dezimalpunkt werden als gültige Eingaben zugelassen).

Zur Übung werden wir nun eine Adreß-Datei aufbauen:

Geben Sie als Bezeichnung des ersten Feldes bitte ein: »NAME« und drücken sie anschließend »Cursor nach unten«. Der Cursor befindet sich nun auf dem ersten Zeichen des zweiten Feldes.

Geben Sie bitte die Zahl »20« als Länge des Feldes mit der Bezeichnung »NAME« ein und bewegen Sie sich zum letzten Feld der Zeile. Geben Sie als Feldtyp bitte nun ein »A« ein.

Dieses »A« steht für ein alphanumerisches Feld und gibt dem Computer an, daß in diesem Feld sowohl Buchstaben als auch Zahlen und Sonderzeichen eingegeben werden dürfen. Betätigen Sie wieder die Taste »Cursor nach unten«.

Die Eingabemaske sollte, falls Sie alle Anweisungen befolgt haben, folgendes Aussehen besitzen:

FELD	BEZEICHNUNG	LÄNGE	TYP
1	NAME	20	A
2			
3			
.			
.			
.			

Sollten Ihnen bei der Eingabe Fehler unterlaufen sein, können Sie sich wie besprochen mit den Cursortasten in das entsprechende Feld begeben und den Fehler korrigieren.

Definieren Sie jetzt die restlichen Datensatzfelder unserer Adreßdatei auf die beschriebene Weise, bis die Maske folgendes Aussehen besitzt:

FELD	BEZEICHNUNG	LÄNGE	TYP
1	NAME	20	A
2	VORNAME	15	A
3	STRASSE	20	A
4	PLZ	4	N
5	ORT	20	A
6	TEL	15	A

Wenn alle Eingaben fehlerfrei sind, drücken Sie bitte die Funktionstaste F1, um die Eingabe zu beenden.

Nun werden Ihnen eventuell aufgetretene Fehler gemeldet. Folgende Fehler sind möglich:

- Sie haben die maximale Datensatzlänge überschritten (245 Zeichen).
- Ihre Eingaben sind unvollständig. Sie haben zum Beispiel bei einem Feld die Bezeichnung angegeben, zum Beispiel Straße, jedoch nicht alle weiteren benötigten Informationen (Länge und Typ).

Im Falle eines solchen Fehlers sehen Sie die entsprechende Fehlermeldung am unteren Bildschirmrand. Korrigieren Sie den Fehler nun bitte und beenden Sie danach wieder mit F1.

Nach der fehlerfreien Definition der einzelnen Datensatzfelder erscheint die Frage »KORREKTUR (J/N)?«. Sollten Sie nun noch Tippfehler oder Ähnliches feststellen, können Sie die Taste »J« betätigen und Ihre Eingabe nochmals korrigieren.

Wenn Sie jedoch mit Ihren Eingaben zufrieden sind, drücken Sie die Taste »N«.

Sie werden nun gefragt, ob Sie auch wirklich eine neue Datei aufbauen wollen. Dies ist Ihre letzte Möglichkeit, Ihren Entschluß zu korrigieren und ohne eine neue Datei aufzubauen, zum Auswahlmenü zurückzukehren.

Da wir wirklich eine Datei aufbauen wollen, betätigen Sie die Taste »J«.

Nun werden Sie aufgefordert, eine leere Diskette in das Laufwerk einzulegen. Die Diskette, die Sie einlegen, muß bereits formatiert sein.

Bedenken Sie bei der Auswahl der Diskette bitte, daß alle Informationen, die sich noch darauf befinden, beim Dateiaufbau gelöscht werden.

Betätigen Sie nun eine beliebige Taste (jedoch nicht »RUN/STOP«). Die Datei wird nun aufgebaut, ein Vorgang, der mehrere Minuten in Anspruch nimmt.

Wichtig!

Das erste Feld (Feld Nr. 1), das Sie beim Dateiaufbau definieren, ist ein »Index-Feld«. Ein solches Index-Feld ermöglicht eine besonders schnelle Suche nach Datensätzen, da die Datensätze nach diesem Feld gewissermaßen geordnet sind. Geben Sie daher als erstes Feld jenes ein, über das Sie bei der späteren Arbeit mit der Datei am häufigsten nach Datensätzen suchen werden.

Da wir in unserer Beispieldatei als erstes das Feld »NAME« angegeben haben, wird die spätere Suche nach Datensätzen über dieses Feld am schnellsten vor sich gehen.

Datendiskette wechseln

Wenn Sie bereits mehrere Dateien aufgebaut haben, können Sie mit Hilfe dieser Funktion von einer Datei, die Sie gerade bearbeiten, auf eine andere Datei überwechseln.

Wählen Sie diese Funktion an. Sie werden nun aufgefordert, die Datendiskette mit der Datei, die Sie nun bearbeiten wollen, einzulegen.

Drücken Sie bitte eine beliebige Taste, nachdem Sie dies getan haben. Sie können nun mit dieser zweiten Datei arbeiten.

Eintragen von Datensätzen

Wählen Sie diesen Programmpunkt an, um in die von Ihnen aufgebaute Datei Datensätze einzutragen.

Auf dem Bildschirm sehen Sie eine Eingabemaske, die gemäß Ihren beim Dateiaufbau gemachten Angaben gestaltet ist. Wenn Sie den jeweiligen Datensatz, zum Beispiel eine Adresse, in die Maske eingegeben haben, drücken Sie bitte F1. Die Daten werden nun gespeichert.

Suchen von Datensätzen

Diese Funktion ist besonders wichtig, da Sie sie wahr-

scheinlich am häufigsten verwenden werden.

Prinzipiell gibt es zwei Formen der Suche nach Daten:
- die Suche über den Index (das erste Feld der Maske).
- die Suche über andere als das Index-Feld.

In beiden Fällen dürfen Sie in beliebig viele Felder Suchkriterien eingeben. Wenn Sie in unserer Beispieldatei den Herrn »MÜLLER« in »MANNHEIM« suchen, geben Sie ein:

```
NAME:     MÜLLER
VORNAME:
STRASSE:
PLZ:
ORT:      MANNHEIM
TEL:
```

Da eine Eingabe in das erste Feld, das Index-Feld erfolgte, wird die Suche recht schnell vor sich gehen. In dem Beispiel haben wir zwei Suchkriterien eingegeben. Es können beliebig viele Suchkriterien eingegeben werden, maximal eines in jedem Feld.

Auf dem Bildschirm werden dabei nur jene Datensätze ausgegeben, die allen eingegebenen Suchkriterien entsprechen, also in unserem Beispiel nicht der Herr »MÜLLER« in »MÜNCHEN«, da dieser dem in das Feld »ORT« eingegebenen Suchkriterium nicht entspricht.

Wenn eines der Suchkriterien in das erste Feld, also das Index-Feld eingegeben wurde, verläuft die Suche schneller, als wenn das nicht der Fall ist.

Die Suche kann jederzeit durch Drücken einer beliebigen Taste abgebrochen werden.

Nachdem ein Datensatz gefunden wurde, der den eingegebenen Kriterien entspricht, kann dieser ausgedruckt werden, indem die betreffende Frage durch Drücken von »J« beantwortet wird. Wenn Sie keinen Ausdruck wünschen, drücken Sie bitte »N«.

Wenn Sie auf die nun auftauchende Frage »WEITERBLÄTTERN (J/N)?« mit »J« antworten, wird nach dem nächsten Datensatz gesucht, der allen eingegebenen Kriterien entspricht.

Wenn das Dateiende erreicht wurde, kehren Sie durch Drücken einer beliebigen Taste zum Auswahlmenü zurück.

Abkürzen

Es ist möglich, einzugebende Suchkriterien beliebig abzukürzen.

Beispiel:

```
NAME:     SCHMID
VORNAME:
STRASSE:
PLZ:
ORT:      MA
TEL:
```

findet:
- SCHMIDT in MANNHEIM
- SCHMIDTZ in MANNHEIM
- SCHMIDKE in MAINZ

jedoch nicht:
- SCHMIT in MANNHEIM
- SCHMOLL in MAINZ
- SCHMIDT in MÜNCHEN

Achtung: Wenn die Eingabe in das erste Feld, also das Index-Feld, auf weniger als sechs Zeichen abgekürzt wird, dient dieses nicht mehr als Index-Feld, das heißt die Suche verläuft langsamer!

Wenn Sie Wert auf eine schnelle Suche legen, kürzen Sie daher eine Eingabe in das erste Feld möglichst nicht auf weniger als diese sechs Zeichen ab.

Ändern von Datensätzen

Zum Ändern von Datensätzen müssen diese zuerst gefunden werden. Geben Sie daher bitte Ihre Suchkriterien in die Maske ein. Wenn der gesuchte Datensatz gefunden wurde, werden Sie gefragt, ob dieser gelöscht werden soll; ein zum Ändern notwendiger Vorgang. Wenn der auf dem Bildschirm ausgegebene Datensatz nicht der Gesuchte ist, geben Sie auf diese Frage ein »N« ein und blättern Sie bis zum gesuchten Datensatz weiter.

Nachdem der gesuchte Datensatz gelöscht wurde, können Sie ihn nun nach Ihren Wünschen ändern.

Durch Drücken der Taste F1 wird der geänderte Datensatz gespeichert.

Löschen von Datensätzen

Auch bei dieser Funktion muß der zu löschende Datensatz zuerst gefunden werden.

Geben Sie Ihre Suchkriterien ein und drücken Sie F1. Wenn ein ausgegebener Datensatz nicht der gesuchte ist, geben Sie auf die Frage »LÖSCHEN (J/N)?« bitte »N« ein und blättern Sie weiter bis zum gesuchten Datensatz.

Blättern in der Datei

Sollten Sie einen bestimmten Datensatz suchen, jedoch keinerlei verwendbare Suchkriterien kennen, können Sie sich mit Hilfe dieser Funktion alle in der Datei gespeicherten Datensätze ansehen.

Dieses Durchblättern der Datei kann jederzeit von Ihnen abgebrochen werden, indem Sie eine beliebige Taste drücken.

Komplette Datei ausdrucken

Wenn Sie diese Funktion anwählen (und Ihren Drucker angeschlossen und eingeschaltet haben), wird die komplette Datei ausgedruckt.

Auch diese Funktion können Sie jederzeit durch Drücken einer beliebigen Taste abbrechen.

Verlassen des Programms

Obwohl es bereits erwähnt wurde, möchte ich noch einmal darauf hinweisen, daß Sie die Arbeit mit dem Programm nur mit Hilfe dieser Funktion beenden dürfen, wenn Sie nicht die Gefahr eines Datenverlustes eingehen wollen.

(S. Baloui/tr)

Maximale Dateianzahl: beliebig (eine Datei pro Diskette)
Maximale Datensatzanzahl pro Datei: 662
Maximale Datensatzlänge: 245 Zeichen
Maximale Feldanzahl pro Datensatz: 10
Maximale Feldlänge: 67 Zeichen
Suchprinzipien: 2 (Suche über Index. Suche ohne Index)
Suchkriterien: Maximal 10 (eines pro Feld)

Tabelle 1. Technische Daten von »DATAMASTER«



```
5 FORA=1TO8:KEYA,"":NEXT
6 KEY8,"Z":KEY1,"S"
7 COLOR1,15,5:COLOR0,7,0:COLOR4,7,0
10 CS=202
20 CZ=205
30 CR=55464
40 SC=3072
50 PA=-1024
60 TZ=239
70 TP=1319
80 GOSUB4530
90 DIMS%(30),Z%(30),BZ$(30),LE%(30),TY$(
30),UG$(30),BZ%(82)
100 CLOSE2:CLOSE15
110 OPEN15,8,15,"I0"
120 GOSUB150
130 GOTO670
140 :
150 REM *DATEI EINLESEN*
160 GOSUB380
170 PRINT#15,"U1:"2;0;18;0:PRINT#15,"B-P
:"2;162:REM ID
180 GET#2,A$:GET#2,B$:ID$=A$+B$
190 IFID$<>"DD"THENGOSUB390:RETURN
200 RN=663:GOSUB410
210 GOSUB360:GOSUB350
220 INPUT#2,AD,FZ
230 MF=FZ
240 FORA=1TOFZ
250 S%(A)=0:Z%(A)=1+A*2
260 INPUT#2,BZ$(A),LE%(A),TY$(A)
270 NEXT
280 RN=664:GOSUB410
290 GOSUB360:GOSUB350
300 FORA=0TO82:INPUT#2,B$:BZ%(A)=VAL(B$)
:NEXT
310 GOSUB390
320 RETURN
330 :
340 REM *B-P,U1,U2,OPEN,CLOSE*
350 PRINT#15,"B-P:";2;1:RETURN
360 PRINT#15,"U1:";2;0;RT;RS:RETURN
370 PRINT#15,"U2:";2;0;RT;RS:RETURN
380 OPEN2,8,2,"#":RETURN
390 CLOSE2:RETURN
400 :
410 REM *BLOCKUMRECHNUNG*
420 IFRN<358THENAA=0:BB=22:DD=1:GOTO460
430 IFRN<471THENAA=357:BB=20:DD=19:GOTO4
60
440 IFRN<580THENAA=471:BB=19:DD=25:GOTO4
60
450 AA=579:BB=18:DD=31
460 RT=INT(((RN-AA)-1)/(BB-1))+DD:RS=RN-
AA-(RT-DD)*BB+(RT-DD-1):RETURN
470 :
480 REM *HASH-ZAHL*
490 HZ$=UG$(1)+"AAAAAA"
500 H1=0:C=0
510 FORA=1TO6
520 H1=ASC(MID$(HZ$,A,1))
530 H1=(H1ORC)-(H1ANDC)
540 C=((2*H1)AND255)OR(SGN(CAND128))
550 NEXT
560 HZ=INT(H1*662/255):IFHZ=0THENHZ=1
570 RETURN
580 :
590 REM *MASKENDEFINITION*
600 FORA=1TOFZ
610 S%(A)=0:Z%(A)=1+A*2
620 BZ$(A)=FB$(A):LE%(A)=FL(A):TY$(A)=FT
$(A)
630 NEXT
640 MF=FZ
650 RETURN
660 :
670 REM *MENUE*
680 PRINT"{CLR}"
690 GOSUB1830:PRINT"{RVSON,SPACE,RVOFF}"
;
700 PRINT"{RVSON,5SPACE}AUSWAHLMENUE{23S
PACE,RVOFF}":PRINT
710 PRINT"{3SPACE,RVSON}D{RVOFF }DATENDI
SKETTE WECHSELN"
720 PRINT"{DOWN,3SPACE,RVSON}E{RVOFF }EI
NTRAGEN VON DATENSAETZEN"
730 PRINT"{DOWN,3SPACE,RVSON}S{RVOFF }SU
CHEN VON DATENSAETZEN"
740 PRINT"{DOWN,3SPACE,RVSON}A{RVOFF }AE
NDERN VON DATENSAETZEN"
750 PRINT"{DOWN,3SPACE,RVSON}L{RVOFF }LO
ESCHEN VON DATENSAETZEN"
760 PRINT"{DOWN,3SPACE,RVSON}B{RVOFF }BL
AETTERN IN DER DATEI"
770 PRINT"{DOWN,3SPACE,RVSON}K{RVOFF }KO
MPLETTE DATEI AUSDRUCKEN"
780 PRINT"{DOWN,3SPACE,RVSON}N{RVOFF }NE
UE DATEI AUFBAUEN"
790 PRINT"{DOWN,3SPACE,RVSON}V{RVOFF }VE
RLASSEN DES PROGRAMMS"
800 PRINT
810 GOSUB1830:PRINT"{RVSON,SPACE,RVOFF}"
;
820 PRINT"{RVSON,5SPACE}KOMMANDO ?{25SPA
CE,RVOFF}";
830 FR=FRE(0)
840 K$="DESALBKN"
850 GETA$:FORA=1TOLEN(K$)
860 IFA$=MID$(K$,A,1)THEN910
870 IFA$="V"ANDID$="DD"THENGOSUB4370
880 IFA$="V"THENCLOSE15:END
890 NEXT
900 GOTO850
910 IFID$<>"DD"ANDA$<>"N"ANDA$<>"D"THEN8
50
920 ONAGOSUB3310,1860,2190,3030,3110,317
0,3230,3390,4370
930 GOTO670
940 :
950 REM *FUNKTIONSTASTEN*
960 POKECS,0:POKECZ,23:SYSCR
970 PRINT"{RVSON,3SPACE}F1 : DURCHFUEHRU
NG DER FUNKTION{5SPACE,RVOFF}"
980 PRINT"{RVSON }HELP : RUECKKEHR ZUM A
USWAHLMENUE{5SPACE,RVOFF}";
990 RETURN
1000 :
1010 REM *MASKENAUFBAU/STEUERUNG*
1020 RF=0:REM RETURNFLAG INIT.
1030 FORA=1TOMF
1040 POKECS,S%(A):POKECZ,Z%(A):SYSCR
1050 IFDA=0THENPRINTBZ$(A)"?"
1060 NEXT
1070 ZZ=1
1080 :
1090 :
1100 :
1110 IFDA=1THENPOKECS,S%(ZZ)+LEN(BZ$(ZZ)
)+1:GOTO1130
1120 POKECS,S%(ZZ)+LEN(BZ$(ZZ))+2
1130 POKECZ,Z%(ZZ):SYSCR
1140 GOSUB1410:REM EINGABEROUTINE
```

Listing »Datamaster« für den C16. Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
1150 POKEA,PEEK(A)AND127
1160 IFA$="{UP}"ANDZZ>1THENZZ=ZZ-1:GOTO1
080
1170 IF(A$="{DOWN}"ORASC(A$)=13)ANDZZ<MF
THENZZ=ZZ+1:GOTO1080
1180 IFA$="Z"THENRF=1:RETURN
1190 IFA$<>"S"THEN1080
1200 :
1210 :
1220 :
1230 IFDA<>1THEN1340
1240 DA=0:FORA=1TOMF
1250 UG$(A)=""
1260 SP=40*Z%(A)+S%(A)+LEN(BZ$(A))+SC+1
1270 FORB=SPTOSP+LE%(A)-1
1280 PE=PEEK(B):IFPE<32THENPE=PE+64
1290 UG$(A)=UG$(A)+CHR$(PE)
1300 NEXTB,A
1310 RETURN
1320 :
1330 REM *MASKENFELDER EINLESEN*
1340 FORA=1TOMF
1350 UG$(A)=""
1360 POKECS,10:POKECZ,Z%(A):SYSCR
1370 POKETZ,1:POKETP,13:INPUTUG$(A)
1380 NEXT
1390 RETURN
1400 :
1410 REM *EINGABEROUTINE*
1420 ML=0
1430 A=SC+80
1440 POKEA,PEEK(A)AND127
1450 A=SC+40*PEEK(CZ)+PEEK(CS)
1460 POKEA,PEEK(A)OR128:POKEA+PA,93
1470 GETA$:IFA$=""THEN1470
1480 IFML=LE%(ZZ)THEN1570
1490 IFTY$(ZZ)="A"THEN1520
1500 IFTY$(ZZ)="N"THEN1540
1510 IFTY$(ZZ)="B"THEN1560
1520 IFASC(A$)=34ORA$=":"ORA$=","ORA$<"
"ORA$>"Z"THEN1570
1530 ML=ML+1:PRINTA$;:GOTO1440
1540 IFA$>","ANDA$<":"ORA$=" "THENML=ML+
1:PRINTA$;:GOTO1440
1550 GOTO1570
1560 IFA$="A"ORA$="N"ORA$=" "THENML=ML+1
:PRINTA$;:GOTO1440
1570 IFASC(A$)=20ANDDA=0ANDML>0THENML=ML
-1:PRINTA$;:GOTO1440
1580 IFA$="{LEFT}"ANDML>0THENML=ML-1:PRI
NTA$;:GOTO1440
1590 IFA$="{RIGHT}"ANDML<LE%(ZZ)THENML=M
L+1:PRINTA$;:GOTO1440
1600 IFA$="{DOWN}"ORA$="{UP}"ORA$="Z"ORA
$="S"THENRETURN
1610 GOTO1440
1620 :
1630 REM *INFOS DATEIAUFBAU*
1640 PRINT"{RVSON}FELDNR{4SPACE}BEZEICHN
UNG{4SPACE}LAENGE{4SPACE}TYP "
1650 PRINT"{RVSON,11SPACE}(MAX.10){5SPAC
E}(MAX.67){2SPACE}(A/N){RVOFF}"
1660 RETURN
1670 :
1680 REM *INFOS 3*
1690 IFAF=1THENPRINT"{RVSON,5SPACE}AENDE
RN VON DATENSAETZEN{11SPACE,RVOFF}";:RET
URN
1700 IFBF=1THENPRINT"{RVSON,5SPACE}BLAET
TERN IN DER DATEI{13SPACE,RVOFF}";:RETUR
N
1710 IFEF=1THENPRINT"{RVSON,5SPACE}EINTR
AGEN VON DATENSAETZEN{9SPACE,RVOFF}";:RE
TURN
1720 IFLF=1THENPRINT"{RVSON,5SPACE}LOESC
HEN VON DATENSAETZEN{10SPACE,RVOFF}";:RE
TURN
1730 PRINT"{RVSON,5SPACE}SUCHEN VON DATE
NSAETZEN{12SPACE,RVOFF}";:RETURN
1740 :
1750 REM *INFOS 4*
1760 POKECS,0:POKECZ,23:SYSCR
1770 PRINT"{RVSON,3SPACE}DATEIENDE ERREI
CHT{18SPACE,RVOFF}"
1780 PRINT"{RVSON,3SPACE}DRUECKEN SIE EI
NE TASTE{13SPACE,RVOFF}";
1790 GETA$:IFA$=""THEN1790
1800 RETURN
1810 :
1820 REM *INFOS 2*
1830 PRINT"{RVSON,39SPACE,RVOFF}";
1840 RETURN
1850 :
1860 REM *EINTRAGEN*
1870 IFID$<>"DD"THENRETURN:REM FALSCHE D
ISKETTE
1880 PRINT"{CLR}";
1890 GOSUB1830:PRINT"{RVSON,SPACE,RVOFF}
";
1900 EF=1:REM EINTR.FLAG SETZEN
1910 GOSUB1680
1920 EF=0:REM EINTR.FLAG LOESCHEN
1930 GOSUB950:GOSUB1010
1940 IFAD=662ORRF=1THENRETURN
1950 AD=AD+1
1960 OPEN2,8,2,"#"
1970 GOSUB480:REM HASH-ZAHL
1980 RN=HZ
1990 PRINT"{HOME,RVSON}"RN" "
2000 BY=INT((RN-1)/8):BI=RN-1-8*BY
2010 IF(BZ%(BY)AND2↑BI)<>0THEN2060
2020 GOSUB410
2030 GOSUB360:GOSUB350
2040 GET#2,A$
2050 IFA$="↑"THEN2090
2060 RN=RN+1:IFRN=663THENRN=1
2070 IFRN=HZTHENCLOSE2:RETURN
2080 GOTO1990
2090 GOSUB350
2100 FORA=1TOFZ
2110 IFUG$(A)=""THENUG$(A)="*"
2120 PRINT#2,UG$(A);CHR$(13);
2130 NEXT
2140 BZ%(BY)=BZ%(BY)OR2↑BI
2150 GOSUB370:CLOSE2
2160 IFAF=1ORLF=1THENRETURN
2170 GOTO1880
2180 :
2190 REM *SUCHEN*
2200 RF=0:REM RETURNFLAG INIT.
2210 HF=0:REM HILFSFLAG INITIALISIEREN (
BLOCKGRENZE UEBERSCHRITTEN?)
2220 PRINT"{CLR}";
2230 GOSUB1830:PRINT"{RVSON,SPACE,RVOFF}
";
2240 GOSUB1680
2250 IFBF=1THEN2280
2260 GOSUB950:GOSUB1010
2270 IFRF=1THENRETURN
2280 OPEN2,8,2,"#"
2290 GOSUB480:REM HASH-ZAHL
2300 RN=HZ:GOSUB2830:REM SUCHABBRUCH
```

```
2310 PRINT"{HOME,RVSON}"RN" "
2320 BY=INT((RN-1)/8):IFBZ%(BY)=0THENRN=
8*(BY+1)+1:GOTO2380
2330 BI=RN-1-8*BY:IF(BZ%(BY)AND2↑BI)=0TH
EN2370
2340 GOSUB410
2350 GOSUB360:GOSUB350
2360 GET#2,A$:IFA$<>"↑"THEN2420
2370 RN=RN+1
2380 IFRN>662THENRN=1:HF=1:REM HF=1:BLOC
KGRENZE UEBERSCHRITTEN
2390 IFHF=1THENIFRN>=HZTHENGOSUB1750:RF=
1:CLOSE2:RETURN
2400 GETA$:IFA$=""THEN2310
2410 RF=1:CLOSE2:RETURN
2420 GOSUB350
2430 FORA=1TOFZ:INPUT#2,AG$(A):IFAG$(A)=
"*"THENAG$(A)=""
2440 NEXT
2450 IFBF=1THEN2500
2460 FORB=1TOFZ
2470 IFUG$(B)=""THEN2490
2480 IFLEFT$(AG$(B),LEN(UG$(B)))<>UG$(B)
THEN2370
2490 NEXT
2500 PRINT"{CLR}";
2510 GOSUB1830:PRINT"{RVSON,SPACE,RVOFF}
";
2520 GOSUB1680
2530 FORB=1TOFZ
2540 POKECS,0:POKECZ,1+B*2:SYSCR
2550 PRINTBZ$(B)+": ";:PRINTAG$(B)
2560 NEXT
2570 GOSUB2860:REM DRUCKEN?
2580 IFLF=1THENGOSUB2710:REM LOESCHFLAG
GESETZT?
2590 IFLF=1ANDA$="J"THENCLOSE2:RETURN
2600 IFUL=1ANDLF=1THENRF=1:CLOSE2:RETURN
2610 IFUL=1THENGOSUB1750:CLOSE2:RETURN
2620 IFKF=1THEN2690
2630 POKECS,0:POKECZ,23:SYSCR
2640 PRINT"{RVSON,5SPACE}WEITERSUCHEN (J
/N) ?{14SPACE,RVOFF}"
2650 GOSUB1830
2660 GETA$:IFA$<>"J"ANDA$<>"N"THEN2660
2670 IFA$="N"AND(BF=1ORAF=1ORLF=1ORKF=1)
THENCLOSE2:RETURN
2680 IFA$="N"THENCLOSE2:GOTO2200
2690 GOSUB2830:GOTO2370
2700 :
2710 REM *SICHERHEITSABFRAGE*
2720 POKECS,0:POKECZ,23:SYSCR
2730 PRINT"{RVSON,5SPACE}LOESCHEN (J/N)
?{18SPACE,RVOFF}"
2740 GOSUB1830
2750 GETA$:IFA$<>"J"ANDA$<>"N"THEN2750
2760 IFA$="N"THENRETURN
2770 BZ%(BY)=BZ%(BY)ANDNOT2↑BI
2780 GOSUB350:PRINT#2,"↑":GOSUB370
2790 AD=AD-1:GF=1:REM GELOESCHTFLAG SETZ
EN
2800 RETURN
2810 :
2820 REM *SUCHABBRUCH*
2830 POKECS,0:POKECZ,23:SYSCR:PRINT"{RVS
ON }ABBRUCH DER SUCHE MIT BELIEBIGER TAS
TE{RVOFF}"
2840 GOSUB1830:RETURN
2850 :
2860 REM *DRUCKEN*
2870 IFKF=1THEN2920
2880 POKECS,0:POKECZ,23:SYSCR
2890 PRINT"{RVSON,5SPACE}AUSDRUCKEN (J/N
) ?{16SPACE,RVOFF}":GOSUB1830
2900 GETA$:IFA$<>"J"ANDA$<>"N"THEN2900
2910 IFA$="N"THENRETURN
2920 OPEN4,4
2930 FORB=1TOFZ
2940 IFAG$(B)=""THEN2970
2950 IFKF=1THENPRINT#4,AG$(B)"{2SPACE}";
:GOTO2970
2960 PRINT#4,BZ$(B)+" :{4SPACE}"AG$(B)
2970 NEXT
2980 IFKF=1THENPRINT#4
2990 PRINT#4
3000 CLOSE4
3010 RETURN
3020 :
3030 REM *AENDERN*
3040 AF=1:REM AENDERNFLAG SETZEN
3050 GOSUB3110:REM LOESCHEN
3060 IFRF=1ORGF=0THENAF=0:RETURN:REM RET
URNFLAG GESETZT?
3070 PRINT"{HOME}";:GOSUB1890:REM EINTRA
GEN
3080 AF=0:REM AENDERNFLAG LOESCHEN
3090 RETURN
3100 :
3110 REM *LOESCHEN*
3120 LF=1:REM LOESCHFLAG SETZEN
3130 GOSUB2190:REM SUCHEN
3140 LF=0:REM LOESCHFLAG LOESCHEN
3150 RETURN
3160 :
3170 REM *BLAETTERN*
3180 BF=1:REM BLAETTERNFLAG SETZEN
3190 GOSUB2190:REM SUCHEN
3200 BF=0:REM BLAETTERNFLAG LOESCHEN
3210 RETURN
3220 :
3230 REM *KOMPLETTE DATEI AUSDRUCKEN*
3240 OPEN4,4
3250 A$="********************************
******":PRINT#4,A$
3260 PRINT#4,"*DATAMASTER VERTRIEB:BALOU
I SOFTWARE*"
3270 PRINT#4,A$
3280 FORA=1TO3:PRINT#4:NEXT:CLOSE4
3290 KF=1:GOSUB3170:KF=0:RETURN
3300 :
3310 REM *DATENDISK WECHSELN*
3320 IFID$="DD"THENGOSUB4370
3330 POKECS,0:POKECZ,22:SYSCR
3340 PRINT"{RVSON }BITTE LEGEN SIE EINE
DATENDISKETTE EIN{SPACE,RVOFF}";
3350 PRINT"{RVSON }UND DRUECKEN SIE EINE
 BELIEBIGE TASTE{2SPACE,RVOFF}";
3360 GETA$:IFA$=""THEN3360
3370 GOSUB150:RETURN
3380 :
3390 REM *DATEIAUFBAU*
3400 IFID$="DD"THENGOSUB4370
3410 PRINT"{CLR}";
3420 GOSUB1630
3430 GOSUB950:REM INFOS FUNKTIONSTASTEN
3440 :
3450 REM MASKENDEFINITION
3460 MF=30
3470 FORA=1TO10
3480 POKECS,0:POKECZ,1+2*A:SYSCR
3490 PRINT"{RVSON}"A"{LEFT,SPACE,RVOFF}"
3500 NEXT
```

```
3510 IFFZ=0THEN3550
3520 FORA=1TOFZ
3530 FB$(A)=BZ$(A):FL(A)=LE%(A):FT$(A)=T
Y$(A)
3540 NEXT
3550 FORA=1TO30STEP3
3560 S%(A)=9:Z%(A)=3+2*INT(A/3):BZ$(A)="
":LE%(A)=10:TY$(A)="A"
3570 S%(A+1)=24:Z%(A+1)=3+2*INT(A/3):BZ$
(A+1)="":LE%(A+1)=2:TY$(A+1)="N"
3580 S%(A+2)=34:Z%(A+2)=3+2*INT(A/3):BZ$
(A+2)="":LE%(A+2)=1:TY$(A+2)="B"
3590 NEXT
3600 DA=1:GOSUB1010:DA=0:REM MASKENAUFBA
U/STEUERUNG
3610 IFRF=1THENGOSUB590:RETURN
3620 SS=0:FORA=1TO30STEP3:SS=SS+VAL(UG$(
A+1)):IFVAL(UG$(A+1))>67THEN3640
3630 NEXT:SS=SS+10:IFSS<255THEN3690
3640 POKECS,0:POKECZ,23:SYSCR
3650 PRINT"{RVSON }FELD- BZW. SATZLAENGE
 UEBERSCHRITTEN!{RVOFF}"
3660 PRINT"{RVSON,5SPACE}DRUECKEN SIE EI
NE TASTE{10SPACE,RVOFF}";
3670 GETA$:IFA$=""THEN3670
3680 GOSUB950:DA=1:GOSUB1070:DA=0:GOTO36
10:REM EINSPRG.MASKENSTEUERUNG
3690 FORA=1TO30STEP3
3700 IFUG$(A)="{10SPACE}"THEN3760
3710 IFVAL(UG$(A+1))>0AND(UG$(A+2)="A"OR
UG$(A+2)="N")THEN3760
3720 POKECS,0:POKECZ,23:SYSCR
3730 PRINT"{RVSON}IHRE DATEIBESCHREIBUNG
 IST UNVOLLSTAEN-{RVOFF}"
3740 PRINT"{RVSON }DIG BZW. FEHLERHAFT.
KORRIGIEREN SIE!{SPACE,RVOFF}";:FORB=1TO
3000:NEXTB
3750 GOSUB950:DA=1:GOSUB1070:DA=0:GOTO36
10
3760 NEXTA
3770 POKECS,0:POKECZ,23:SYSCR
3780 PRINT"{RVSON,5SPACE}KORREKTUR (J/N)
 ?{17SPACE,RVOFF}"
3790 GOSUB1830
3800 GETA$:IFA$<>"J"ANDA$<>"N"THEN3800
3810 IFA$="N"THEN3850
3820 GOSUB950:REM INFOS DATEIAUFBAU
3830 DA=1:GOSUB1070:DA=0:GOTO3610:REM EI
NSPRG.MASKENSTEUERUNG
3840 AD=0:REM ANZ.DATENSAETZE INIT.
3850 POKECS,0:POKECZ,23:SYSCR
3860 PRINT"{RVSON,2SPACE}SIND SIE SICHER
, DASS SIE EINE NEUE{2SPACE,RVOFF}"
3870 PRINT"{RVSON,5SPACE}DATEI AUFBAUEN
WOLLEN (J/N) ?{5SPACE,RVOFF}";
3880 GETA$:IFA$<>"J"ANDA$<>"N"THEN3880
3890 IFA$="N"THENGOSUB590:RETURN
3900 FZ=10
3910 FORA=1TO10
3920 FB$(A)=UG$(A*3-2):REM FELDBEZEICHNU
NG
3930 FL(A)=VAL(UG$(A*3-1)):REM FELDLAENG
E
3940 FT$(A)=UG$(A*3):REM FELDTYP
3950 NEXT
3960 FORA=1TO9
3970 IFFB$(A)<>"{10SPACE}"THEN4030
3980 FORB=ATO9
3990 FB$(B)=FB$(B+1):FL(B)=FL(B+1)
4000 FT$(B)=FT$(B+1)
4010 NEXTB
4020 FZ=FZ-1
4030 NEXTA
4040 IFFB$(10)="{10SPACE}"THENFZ=FZ-1
4050 FORA=1TOFZ
4060 S%(A)=0:Z%(A)=1+A*2:BZ$(A)=FB$(A):L
E%(A)=FL(A):TY$(A)=FT$(A)
4070 NEXT
4080 FORA=0TO82:BZ%(A)=0:NEXT
4090 AD=0:REM ANZ.DATENSAETZE INIT.
4100 POKECS,0:POKECZ,23:SYSCR
4110 PRINT"{RVSON}BITTE LEGEN SIE EINE L
EERE DISKETTE EIN{RVOFF}"
4120 PRINT"{RVSON,5SPACE}UND DRUECKEN SI
E EINE TASTE{7SPACE,RVOFF}";
4130 GETA$:IFA$=""THEN4130
4140 GOSUB380:PRINT#15,"U1:"2;0;18;0:PRI
NT#15,"B-P:"2;162
4150 A$="":B$="":GET#2,A$:GET#2,B$:A$=A$
+B$
4160 GOSUB390
4170 IFA$="DD"THEN4100
4180 POKECS,0:POKECZ,23:SYSCR
4190 PRINT"{RVSON,2SPACE}BITTE HABEN SIE
 GEDULD. DER AUFBAU{3SPACE,RVOFF}"
4200 PRINT"{RVSON,2SPACE}DER DATEI BENOE
TIGT MEHRERE MINUTEN{2SPACE,RVOFF}";
4210 PRINT#15,"N:DATAMASTER-DATEI,DD"
4220 GOSUB380:PRINT#15,"U1:"2;0;18;0
4230 ID$="DD":PRINT#15,"B-P:"2;162:PRINT
#2,ID$;:PRINT#15,"U2:"2;0;18;0
4240 PRINT"{HOME,RVSON}BLOCKNR.{RVOFF}"
4250 FORRN=1TO662
4260 PRINT"{HOME,DOWN,RVSON}"RN"{2SPACE,
RVOFF}"
4270 GOSUB410
4280 GOSUB350
4290 PRINT#2,"↑"
4300 GOSUB370
4310 NEXTRN
4320 GOSUB390
4330 GOSUB4370:REM MASKE ABSPEICHERN
4340 GOSUB150:REM DATEN EINLESEN
4350 RETURN
4360 :
4370 REM *VERLASSEN*
4380 GOSUB380
4390 RN=663
4400 GOSUB410
4410 GOSUB350
4420 PRINT#2,AD;CHR$(13);FZ
4430 FORA=1TOFZ
4440 PRINT#2,BZ$(A);CHR$(13);LE%(A);CHR$
(13);TY$(A)
4450 NEXT
4460 GOSUB370
4470 RN=664:GOSUB410:GOSUB350
4480 FORA=0TO82:PRINT#2,STR$(BZ%(A)):NEX
T
4490 GOSUB370:GOSUB390
4500 RETURN
4510 :
4520 REM *TITELBILD*
4530 :
4540 PRINT"{CLR}"
4550 PRINT"{7DOWN,14SPACE}DATAMASTER"
4560 PRINT"{14SPACE}----------"
4570 PRINT"{3DOWN,7SPACE} BALOUI SOFT
WARE, 1985"
4730 RETURN
```

©64'er

Listing »Datamaster« für den C16 (Schluß)

Der VC 20 als Musik Maestro

Mit dem Musik Maestro bringen Sie Ihrem VC 20 Töne bei. Das Programm nutzt die Fähigkeiten des VC 20, so weit es geht, aus. Trotzdem ist das Programm leicht zu bedienen und begeistert alle, die sich bisher nicht mit Musik beschäftigt haben.

Musik Maestro wurde programmiert, um polyphone Musikstücke mit bis zu drei Stimmen spielen zu können. Das Programm wurde zusätzlich mit einer ansprechenden Grafik (Notenschlüssel, Notenzeichen etc.) ausgestattet. Dabei werden alle drei Stimmen gleichzeitig auf dem Bildschirm dargestellt. Die Programmierung der Musikstücke ist »kinderleicht« und wird Ihnen viel Spaß machen.

Im Direktmodus folgende Zeile eingeben:
Laden des Programms:
`POKE 44,32:POKE 43,1:POKE 8192,0:NEW` sowie
`:POKE 648,30: SYS 58648.`

Danach lädt man das Programm »Music Maestro Basic Loader« (Listing 1) und startet es durch RUN. Es bringt das Maschinenspracheprogramm und einen neu definierten Zeichensatz in das unter dem Basic-Anfang liegende RAM. Das Programm ist mit Prüfsummen versehen, so daß bei einem Eingabefehler die fehlerhafte Zeile ausgewiesen wird.

Wenn das Programm beendet ist, fordert es dazu auf, es zu löschen und das Hauptprogramm einzugeben. Man gibt NEW ein und lädt das Hauptprogramm (Music Maestro Basic, Listing 2). Dieses startet man einfach mit RUN.

»Music-Maestro« ist dazu geeignet, polyphone Musikstücke mit bis zu drei Stimmen zu spielen, einzugeben, zu ändern und zu transponieren, oder eigene Musikstücke zu komponieren und abzuspielen. Es zeichnet sich durch äußerst komfortable Eingabe der Noten aus (meist genügen zwei Tasten pro Note), durch vielseitige Editiermöglichkeiten und vor allem durch eine gute Notengrafik (beim Eingeben und beim Spielen). Die Grafik stellt sämtliche Noten völlig korrekt dar mit allen Zusatzzeichen wie Kreuz, Punktierung, Hilfslinien, etc. Es können drei Notenschlüssel gewählt werden sowie sieben verschiedene Tempi. Es lassen sich alle Stimmen zusammen spielen oder auch einzeln. Jede Melodie kann beliebig höher oder niedriger transponiert werden (zum Beispiel von C-Dur nach G-Dur). Schließlich können die eingegebenen Stücke auf Kassette gespeichert und wieder geladen werden. Die Aufteilung des Speichers läßt sich frei wählen. Dank einer codierten Speicherung wird pro Note nur ein Byte Speicherplatz benötigt, so daß trotz der Länge des Programms noch über 5000 Töne zur Verfügung stehen, mit 24 KByte-Erweiterung sogar 13000 Töne.

Bedienung des Programms im einzelnen:

Nach dem Starten beginnt das Programm mit einer einfachen Demonstration (Unterprogramm ab Zeile 32000). Durch Drücken einer beliebigen Taste kommt man ins Menü, wo man zwischen sieben Optionen wählen kann. Diese werden im folgenden beschrieben.

Noten eingeben/editieren:

Hier wählt man zunächst die Stimme aus (1 bis 3). Sodann kann man Vorzeichen (Kreuz oder b) eingeben, die für die ganze Zeit des Eingebens gelten, bis sie bei einem erneuten Sprung vom Menü in den Eingabeteil geändert werden. Danach kann man Darstellung der Noten im Violin-, B- oder Baßschlüssel wählen. Nun erscheint der Bildschirmaufbau. Oben wird die Stimme angezeigt, darunter die Nummer des Tones, der als nächster eingegeben wird, sowie die Zahl der noch freien Töne. Es stehen insgesamt über drei Oktaven zur Verfügung. Die rote senkrechte Linie links zeigt ungefähr an, welche Oktave gerade gewählt wurde. Mit den Tasten F1, F3, F5 und F7 kann von Oktave 1 bis Oktave 4 gewählt werden.

Noten eingeben

Dies geschieht sehr einfach. Will man zum Beispiel das »C« eingeben, so drückt man einfach die Taste »C«, für das »D« die Taste »D« und so weiter, für das »H« allerdings die Taste »B«. Jetzt erscheint auf dem Bildschirm das violette Zeitwertsymbol, das anzeigt, daß nun die Eingabe der Länge der Note erwartet wird (ganze, halbe, viertel und so weiter). Dies geschieht mit den Tasten 1 bis 7. Taste 1: ganze Note, Taste 2: halbe, Taste 3: viertel, Taste 4: achtel und so weiter bis Taste 7: hundertachtundzwanzigstel. Nun wird im Notensystem die Note angezeigt und gespielt und der Zähler für die Ton-Nummer erhöht sich. Jetzt kann die nächste Note eingegeben werden. (Für das »C« wählt man die Oktave 2, für das »C'« die Oktave 3, für das »C"« die Oktave 4 (siehe Bild)). Hat man vorher Vorzeichen eingegeben, zum Beispiel »#« für »F«, so wird ein »F« automatisch zu »Fis« erhöht. Es können natürlich auch alle anderen Noten erhöht, beziehungsweise erniedrigt werden. Dazu gibt man einfach die Note geshiftet oder mit der Commodore-Taste ein. Soll im Beispielsfall das »F« ausnahmsweise nicht erhöht werden, so drückt man erst die CTRL-Taste (es erscheint ein Auflösungszeichen) und danach das »F«. Der Vorteil mit den Oktavenwahltasten liegt darin, daß man solange Noten mit nur zwei Tasten eingeben kann, bis die Oktave gewechselt werden soll, erst danach muß man eine weitere Taste drücken.

Punktierte Noten:

Will man zum Beispiel eine halbe Note punktieren, gibt man erst die halbe Note und dann die gleiche noch einmal als Viertel-Note ein – das Programm erkennt die Punktierung und stellt sie später richtig dar.

Editieren:

Zum Korrigieren und zum Überprüfen der bereits eingegebenen Noten kann man wie folgt editieren: Taste »←« geht einen Schritt zurück und zeigt die hier eingegebene Note an beziehungsweise spielt sie, mit Taste »↑« geht man einen Schritt vorwärts. Will man die so überprüfte Note ändern, gibt man jetzt einfach die gewünschte neue Note ein; die alte Note wird überschrieben. Mit SHIFT F1,F3 beziehungsweise SHIFT F5,F7 kann man zehn und 50 Schritte vor- und zurückgehen. Mit Taste »+« (Insert) werden alle folgenden Töne einen Schritt nach vorne geschoben, so daß eine Note eingefügt werden kann, mit Taste »–« (Delete) wird die zuletzt eingegebene beziehungsweise editierte Note gelöscht, alle folgenden Noten rücken einen Schritt zurück (alle Tasten haben Wiederholautomatik).

Wenn man beim Eingeben oder Editieren über den zulässigen Bereich vor- oder zurückgehen will, ertönt ein neunmaliges Warnsignal, und der Befehl wird nicht ausgeführt. Taste »P« und die Tasten eins bis sieben : Pause (ganze, halbe, viertel und so weiter). Taste »*«: End-Marker. Hierdurch wird das Ende des Musikstücks beim Spielen angezeigt, der Marker kann jederzeit geändert beziehungsweise überschrieben werden; es sollte jedoch immer ein Marker gesetzt werden. Beim Editieren wird der Marker durch dreimaligen Piepton angezeigt. Man kann natürlich über den Marker hinaus editieren und eingeben.

Taste »=«: PLAY (Spielen der eingegebenen Melodie), siehe PLAY.
Taste »£«: Rückkehr zum Menü.

PLAY (Spielen):
Hier wird gefragt, wieviele Stimmen zugleich gespielt werden sollen. Man sollte nur die Stimmen spielen, bei denen man auch Noten eingegeben und den End-Marker gesetzt hat. Dann wird für die gewählten Stimmen gefragt, mit welchem Notenschlüssel sie dargestellt werden sollen. Schließlich kann man das Tempo wählen. Jetzt erscheint nach einigen Sekunden der Bildschirm mit drei verschiedenfarbigen Notensystemen. Durch Drücken einer beliebigen Taste wird das Ganze gestartet.

Während des Abspielens kann man folgendes ändern: Taste F1, F3: schneller, Taste F5, F7: langsamer, Taste »←«: Stop, danach durch Taste »↑« Fortsetzung, Taste »S«: Spielen beenden. Wenn alle Stimmen nicht mehr gespielt werden beziehungsweise wenn Taste »S« gedrückt wurde, kann man mit beliebiger Taste ins Menü zurückkehren (außer »S«), mit Taste »SPACE« wird das Abspielen wiederholt.

Transponieren:
Hierdurch kann man einfach eingegebene Stücke höher oder niedriger versetzen (transponieren), das Programm fragt nach den nötigen Angaben und führt das Transponieren unter akustischer Kontrolle aus.

Reservieren:
Am Anfang werden automatisch für Stimme 1 und 2 je 2000 Töne, für Stimme 3 genau 1000 Töne reserviert. Dies kann man beliebig ändern, wenn zum Beispiel für eine Stimme alle Töne reserviert werden sollen. Das Minimum sind 15 Töne. Zu beachten ist nur, daß nach SAVE oder LOAD wieder die alte Reservierung vom Anfang vorhanden ist, man muß dann gegebenenfalls neu reservieren.

SAVE
Hiermit kann man eingegebene Stücke auf Kassette speichern. Das Programm erfragt die nötigen Daten. Es kann nur jeweils eine Stimme auf einmal gespeichert werden.

Will man also zum Beispiel ein dreistimmiges Stück speichern, so führt man drei SAVE-Operationen für Stimme 1 bis 3 nacheinander aus.

LOAD
Hierdurch wird die gespeicherte Melodie genauso wieder geladen, wie sie gespeichert wurde. Soll die Melodie wieder gespielt werden, muß auch die Reservierung der Töne die gleiche sein wie beim Speichern. Will man zum Beispiel ein dreistimmiges Stück wieder laden, so sind (für die drei Stimmen) drei LOAD-Operationen hintereinander auszuführen. Am besten ist es, wenn man SAVE und LOAD nur mit der vom Programm am Anfang vorgenommenen Reservierung verwendet, wenn man also an der Reservierung nichts ändert.

Variable	
S:	Stimme (0 bis 2)
SF:	Schlüsselflag
0:	Violinschlüssel
1:	B-Schlüssel
2:	Baßschlüssel
OK:	Oktave (0 bis 3)
N:	Zähler für RAM, an welche Adresse eingegebener Ton gespeichert werden soll.
L:	Art der einzugebenden Note (ob C,D,E,F etc.) L=0 : Pause, L=1 : F, L=2 : G, bis L=31 : c"
TP:	Notenwert (ganze, halbe, viertel etc.) (0,1, bis 7)
TA(S), TE(S):	Anfang beziehungsweise Ende der RAM-Adresse, an die Töne der Stimme S zu speichern sind

Geschwindigkeitsänderung:
Wenn man die Geschwindigkeit verlangsamen oder erhöhen will, so kann man den Wert 5 in Zeile 12364 erhöhen (langsamer) beziehungsweise erniedrigen (schneller), ebenso den Wert 35 in derselben Zeile erniedrigen.

(C. Neupert/aw)

Maschinenprogramm-Adressen:

SYS 6067:	Löschen von Stimme S
SYS 5460:	Darstellen der Note für Stimme S
SYS 5490:	Darstellen der Pause für Stimme S
SYS 5120:	PLAY Stimme 0 bis 2

Die Programme benötigen folgende Daten:

für Grafik:

835:	Stimmflag (0 bis 2)
834:	Schlüsselflag (siehe oben SF)
838:	Notenwert (siehe oben TP)
841:	L (siehe oben L)

»PLAY«:

860, 861, 862:	TA(S) LSB für Stimme S
863, 864, 865:	TA(S) MSB für Stimme S
880, 881, 882:	Flag für Stimme S, ob spielen oder nicht (1=ja)

Anmerkungen:
Im Spielmodus werden Halbtöne nur mit Kreuz ausgegeben, nicht mit »b« (aus »g–es« wird also zum Beispiel »fis«), was aber in der Praxis keinen Unterschied bedeutet.
Die letzte obere Note (»c"«) kann nur bis 1/64 Länge haben, 1/128 Länge ergibt den End-Marker.
Punktierte Noten werden korrekt gespielt.

Zusammenfassung der Bedienung des Programms:

Eingabe/Editieren:

Eingabe:	Tasten »C,D,E,F,G,A,B« und Tasten »1 bis 7« (für ganze, halbe, viertel etc.) bis 1/128 Note.	
	Erhöhen:	GeSHIFTete Tasten »C,D« etc.
	Erniedrigen:	Commodore-Taste + C,D etc.
	Auflösungszeichen:	erst CTRL-Taste, dann C,D etc.
	Taste »P« und 1,2,3 etc:	Pause (ganze, halbe, viertel)
	Taste »F1« bis »F7«:	Oktave 1 bis 4
	Taste »*«:	End-Marker setzen
Editieren:	Taste »←«:	1 Ton rückwärts
	Taste »↑«:	1 Ton vorwärts
	SHIFT F1:	wie »↑«, nur 10 Töne
	SHIFT F3:	wie »↑«, nur 50 Töne
	SHIFT F5:	wie »←«, nur 10 Töne
	SHIFT F7:	wie »←«, nur 50 Töne
	Taste »+«:	INSERT (Einfügen)
	Taste »–«:	DELETE (Löschen)
	Taste »=«:	PLAY (Abspielen)
	Taste »£«:	zurück ins Menü
Play:	beliebige Taste, um zu starten.	
	Während des Abspielens:	
	F1, F3:	schneller
	F5, F7:	langsamer
	Taste »←«:	STOP
	Taste »↑«:	GO
	Taste »S«:	Spielen beenden

Nach Beendigung: Taste »SPACE«: wieder spielen
andere Tasten (außer S:) Menü

Anmerkung:
In der untersten Oktave können nur die Töne von »Fis« aufwärts gespielt werden, in der obersten (vierten) Oktave nur das »c"«, andere Noten erzeugen eine Pause.

```
1 REM                                    <063>
2 REM ** MUSIC MAESTRO          **       <047>
3 REM ** MACHINE CODE LOADER **          <041>
4 REM **                                 <053>
5 REM ** BY CLAUS NEUPERT       **       <104>
6 REM **                        **       <006>
7 REM ** BEFORE LOADING:        **       <230>
8 REM ** 'POKE 44,32:POKE 8192,0:NEW' ** <112>
9 REM ** AND:'POKE 648,30:SYS 58648' **  <106>
10 DIM H(75):FOR I=0 TO 9                <139>
20 H(48+I)=I:H(65+I)=I+10:NEXT           <067>
30 FOR I=4750 TO 7675:READ A$            <127>
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1)) <223>
50 D=H(H)*16+H(L):S=S+D:POKE I,D         <243>
60 A=A+1:IF A<20 THEN NEXT: A=-1         <165>
65 PRINT"ZEILE ";1000+Z;                 <181>
70 READ V:Z=Z+1:IF V=S THEN 85           <049>
80 PRINT"FEHLER !";999+Z:STOP            <023>
85 IF A<0 THEN 100                       <223>
90 S=0:A=0:PRINT:NEXT:                   <121>
100 PRINT"{CLR,DOWN}GEBEN SIE 'NEW' EIN" <018>
110 PRINT"{DOWN}UND LADEN SIE"           <194>
120 PRINT"{DOWN}'MUSIC MAESTRO BASIC'"   <221>
130 END                                  <132>
1000 DATA 41,43,48,20,43,4F,44,45,32,32,20
     ,32,32,20,34,38,31,32,38,20, 1078   <220>
1001 DATA 38,20,34,38,31,31,35,36,34,4D,55
     ,53,49,43,20,4D,41,53,54,45, 1248   <080>
1002 DATA AD,40,03,8D,0B,90,8E,40,03,E8,8E
     ,43,03,60,EA,BD,66,03,D0,03, 2024   <006>
1003 DATA 4C,08,16,DE,66,03,60,EA,EA,EA,DE
     ,76,03,A9,01,9D,66,03,60,EA, 2336   <152>
1004 DATA EA,EA,EA,EA,EA,8A,48,0A,0A,0A,18
     ,69,04,48,AA,A0,05,BD,7A,03, 2264   <080>
1005 DATA C9,09,F0,06,FE,7A,03,18,90,0B,A9
     ,00,9D,7A,03,CA,88,C0,00,D0, 2203   <021>
1006 DATA E8,68,A8,68,AA,A9,00,8D,96,03,8E
     ,95,03,98,38,E9,04,AA,BD,7A, 2461   <199>
1007 DATA 03,18,69,60,20,2D,13,E8,EE,96,03
     ,A9,05,CD,96,03,D0,EC,60,48, 2091   <233>
1008 DATA AC,96,03,AD,95,03,C9,00,F0,0E,C9
     ,01,F0,05,68,99,54,1F,60,68, 2124   <026>
1009 DATA 99,AF,1E,60,68,99,19,1E,60,EA,C9
     ,27,D0,10,AE,B6,03,E0,0F,D0, 2366   <064>
1010 DATA 02,A2,FF,E8,8E,B6,03,4C,74,13,C9
     ,2F,F0,01,60,AE,B7,03,E0,07, 2365   <041>
1011 DATA D0,02,A2,FF,E8,8E,B7,03,EA,EA,48
     ,AD,B6,03,0A,0A,0A,0A,B8,6D, 2418   <087>
1012 DATA B7,03,18,69,08,8D,0F,90,68,60,EA
     ,EA,EA,A5,C5,C9,40,D0,01,60, 2457   <099>
1013 DATA C9,27,D0,0C,CE,6C,03,CE,6C,03,CE
     ,6C,03,EA,EA,EA,C9,37,D0,0C, 2589   <160>
1014 DATA EE,6C,03,EE,6C,03,EE,6C,03,EA,EA
     ,EA,C9,08,D0,0C,A5,C5,C9,40, 2805   <181>
1015 DATA D0,FA,A5,C5,C9,36,D0,FA,C9,29,F0
     ,01,60,A2,02,A9,01,9D,53,03, 2689   <017>
1016 DATA 9D,0A,90,CA,EA,EA,10,F5,60,EA,AE
     ,43,03,BD,76,03,F0,0B,AC,44, 2617   <249>
1017 DATA 03,A9,60,99,05,1A,99,06,1A,60,20
     ,EA,15,A5,C5,C9,40,F0,FA,C9, 2338   <104>
1018 DATA 20,F0,07,20,4C,13,18,90,ED,EA,A0
     ,02,B9,70,03,99,56,03,49,01, 1823   <146>
1019 DATA 99,53,03,A9,00,99,76,03,88,10,ED
     ,A0,00,8C,43,03,C8,CC,53,03, 1931   <252>
1020 DATA F0,1E,CC,56,03,D0,19,A2,00,20,6B
     ,1C,A2,00,20,13,15,20,37,15, 1467   <143>
1021 DATA 8D,0C,90,AD,73,03,8D,42,03,20,C5
     ,12,A0,01,8C,43,03,CC,54,03, 1707   <231>
1022 DATA F0,1E,CC,57,03,D0,19,A2,01,20,6B
     ,1C,A2,01,20,13,15,20,37,15, 1470   <175>
1023 DATA 8D,0B,90,AD,74,03,8D,42,03,20,C5
     ,12,A0,02,8C,43,03,88,CC,55, 1842   <009>
1024 DATA 03,D0,0B,CC,54,03,D0,24,CC,53,03
     ,D0,1F,60,CC,58,03,D0,19,A2, 2072   <030>
1025 DATA 02,20,6B,1C,A2,02,20,13,15,20,37
     ,15,8D,0A,90,AD,75,03,8D,42, 1308   <217>
1026 DATA 03,20,C5,12,AD,53,03,C9,01,F0,1D
     ,AD,50,03,CD,4D,03,D0,12,A2, 1909   <044>
1027 DATA 01,8E,56,03,CA,8E,50,03,20,E6,1D
     ,20,AC,17,18,90,03,EE,50,03, 1669   <116>
1028 DATA AD,54,03,C9,01,F0,1D,AD,51,03,CD
     ,4E,03,D0,12,A2,01,8E,57,03, 1895   <112>
1029 DATA CA,8E,51,03,20,B6,12,20,AC,17,18
     ,90,03,EE,51,03,AD,55,03,C9, 1842   <188>
1030 DATA 01,F0,1D,AD,52,03,CD,4F,03,D0,12
     ,A2,01,8E,58,03,CA,8E,52,03, 1866   <104>
1031 DATA 20,63,15,20,AC,17,18,90,03,EE,52
     ,03,20,88,13,EA,EA,EA,EA,20, 2028   <211>
1032 DATA EA,15,4C,15,14,A9,00,9D,56,03,AD
     ,5A,03,9D,4D,03,C9,FF,D0,06, 1960   <160>
1033 DATA A9,01,9D,53,03,60,AD,49,03,C9,00
     ,D0,03,20,72,15,A8,B9,DE,1C, 1940   <043>
1034 DATA 60,48,A9,01,DD,76,03,D0,05,20,D4
     ,12,68,60,AC,49,03,CC,62,03, 1908   <082>
1035 DATA F0,02,68,60,AC,46,03,C8,CC,63,03
     ,F0,02,68,60,9D,76,03,68,9D, 2174   <096>
1036 DATA 3F,03,60,EA,EA,AD,41,03,8D,0A,90
     ,8E,41,03,E8,E8,8E,43,03,60, 2148   <244>
1037 DATA AC,46,03,AE,43,03,BD,C5,1D,18,79
     ,B0,1D,AA,98,B8,C9,03,30,32, 2062   <037>
1038 DATA 38,E9,02,0A,0A,0A,A8,8C,4B,03,A0
     ,00,B9,74,1D,1D,00,19,9D,00, 1408   <046>
1039 DATA 19,E8,C8,CC,4B,03,D0;F0,B8,C0,19
     ,10,10,A9,01,A0,00,1D,00,19, 2004   <224>
1040 DATA 9D,00,19,E8,C8,C0,07,D0,F4,60,C9
     ,02,F0,09,A8,A9,08,8D,4B,03, 2377   <035>
1041 DATA 18,90,06,A8,A9,28,8D,4B,03,98,0A
     ,0A,0A,A8,18,6D,4B,03,8D,4B, 1547   <123>
1042 DATA 03,B9,38,1D,1D,00,19,9D,00,19,E8
     ,C8,CC,4B,03,D0,F0,EA,EA,60, 2235   <066>
1043 DATA A9,00,AA,A8,E8,C8,CC,6C,03,D0,FA
     ,A0,00,20,88,13,EC,6D,03,D0, 2615   <012>
1044 DATA EF,60,EA,60,EA,EA,EA,EA,EA,EA,AE
     ,49,03,BC,10,1D,98,0A,0A,0A, 2734   <196>
1045 DATA 4A,AE,43,03,18,7D,C5,1D,48,AD,42
     ,03,0A,0A,85,02,68,18,65,02, 1393   <042>
1046 DATA AA,8E,44,03,98,4A,4A,4A,4A,4A,8D
     ,45,03,A0,00,8C,4A,16,B8,AD, 1871   <119>
1047 DATA 46,03,C9,02,30,09,A9,08,18,6D,4A
     ,16,8D,4A,16,B9,08,1D,1D,00, 1227   <162>
1048 DATA 19,9D,00,19,E8,C8,C0,08,D0,F1,EA
     ,EA,A9,00,CD,46,03,D0,03,18, 2438   <041>
1049 DATQ 90,4A,AD,42,03,0A,18,69,10,C9,10
     ,D0,03,18,69,01,B8,CD,49,03, 1638   <246>
1050 DATA 30,11,A9,00,8D,3E,03,AD,44,03,38
     ,E9,18,AA,A9,01,18,90,0E,A9, 1688   <152>
1051 DATA 01,8D,3E,03,AD,44,03,18,69,03,AA
     ,A9,80,8D,4A,03,A0,00,8E,47, 1641   <246>
1052 DATA 03,1D,00,19,9D,00,19,AD,4A,03,C8
     ,E8,C0,1C,D0,F1,4C,02,17,EA, 1925   <230>
1053 DATA AC,46,03,B8,C0,03,10,03,EA,EA,60
     ,EA,88,88,AD,3E,03,C9,01,F0, 2393   <243>
1054 DATA 1D,AE,47,03,A9,FF,9D,00,1A,E8,9D
     ,00,1A,E8,9D,00,1A,E8,E8,E8, 2410   <238>
1055 DATA 88,C0,00,D0,ED,60,EA,EA,EA,EA,AD
     ,47,03,18,69,1B,AA,A9,FF,9D, 2959   <206>
1056 DATA 00,19,CA,9D,00,19,CA,9D,00,19,CA
     ,CA,CA,88,C0,00,D0,ED,60,00, 2268   <217>
1057 DATA AD,49,03,85,01,EA,EA,EA,AC,42,03
     ,B9,9F,1D,B8,C5,01,10,2F,B9, 2329   <187>
1058 DATA A2,1D,C5,01,30,03,18,90,5A,A9,01
     ,8D,48,03,A5,01,38,F9,A2,1D, 1746   <035>
1059 DATA AA,CA,BD,70,1D,A8,20,96,17,AA,20
     ,82,17,8A,38,E9,08,AA,88,C0, 2363   <068>
1060 DATA 00,D0,F3,18,90,35,A6,01,E0,00,D0
     ,03,18,90,2C,C0,00,F0,10,C0, 2126   <247>
1061 DATA 01,F0,06,BD,C9,1C,18,90,09,BD,B7
     ,1D,18,90,03,BD,D4,1C,A8,20, 2043   <228>
1062 DATA 96,17,18,69,30,AA,20,82,17,8A,18
     ,69,08,AA,88,C0,00,D0,F3,20, 1961   <180>
1063 DATA C9,1D,20,D8,13,4C,B2,16,A9,07,9D
     ,00,18,A9,FF,9D,00,19,A9,E0, 2129   <074>
1064 DATA 9D,00,1A,60,EA,EA,EA,EA,AD,43,03
     ,C9,00,F0,0A,C9,01,F0,03,A9, 2523   <161>
1065 DATA B4,60,A9,5C,60,A9,0C,60,33,32,A9
     ,00,DD,76,03,D0,4C,AD,43,03, 2049   <248>
1066 DATA C9,00,F0,12,C9,01,F0,07,A2,A8,A0
     ,58,18,90,0B,A2,50,A0,58,18, 2179   <027>
1067 DATA 90,04,A2,00,A0,50,8A,48<A9,00,9D
     ,00,19,9D,00,18,9D,00,1A,88, 1611   <170>
```

Listing 1. Ladeprogramm zu »MUSIC MAESTRO«. Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
1068 DATA E8,C0,00,D0,F1,A0,00,68,18,69,14
     ,AA,A9,FF,9D,00,18,9D,00,19, 2243   <206>
1069 DATA 9D,00,1A,8A,18,69,08,AA,C8,C0,05
     ,D0,EB,60,00,00,00,00,00,00, 1564   <161>
1070 DATA 00,00,00,00,00,00,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 255    <035>
1071 DATA 00,00,FF,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,FF,00, 765    <239>
1072 DATA 00,00,00,00,00,00,FF,00,00,00
     ,00,00,00,00,00,00,00,00,00, 255    <037>
1073 DATA 00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <063>
1074 DATA 00,00,00,00,00,00,07,00,00,00
     ,FB,22,22,22,22,23,00,00,D1, 638    <204>
1075 DATA D9,55,55,53,D3,00,00,F7,84,E7,85
     ,85,84,00,00,BD,A1,BD,21,21, 2294   <230>
1076 DATA BD,00,00,F4,95,96,95,95,F4,00,00
     ,BE,08,08,08,08,89,00,24,7E, 1795   <072>
1077 DATA 24,24,FF,24,7E,24,20,38,26,22,FF
     ,24,38,00,00,00,48,48,FF,48, 1503   <056>
1078 DATA FC,48,48,FC,48,48,FF,48,00,00,40
     ,40,40,40,FF,70,48,44,48,48, 2026   <236>
1079 DATA 70,70,FF,00,00,00,00,00,00,FF
     ,20,20,20,FB,80,80,F8,08,08, 1601   <131>
1080 DATA F8,00,E2,80,80,80,80,80,92,00,FE
     ,80,80,FC,80,80,FE,00,3C,7E, 2718   <159>
1081 DATA FE,FF,FE,BC,80,80,7E,7E,40,7E,7E
     ,40,40,40,80,80,FC,84,84,FC, 2991   <185>
1082 DATA 04,04,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 8      <142>
1083 DATA 00,00,00,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,FF,FF, 765    <153>
1084 DATA FF,FF,FF,FF,00,00,FF,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 1530   <253>
1085 DATA 00,00,FF,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 255    <112>
1086 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,3C,7E,FF,FF,FF, 951    <081>
1087 DATA FE,BC,FF,FF,FF,80,80,FF,FF,FF,FF
     ,80,FF,FF,FF,80,80,FF,FF,FF, 4397   <164>
1088 DATA 80,80,FF,FF,FF,00,FF,00,00,00,00
     ,00,00,00,FF,00,00,00,3C,42, 1657   <182>
1089 DATA 46,5A,62,42,3C,00,08,18,28,08,08
     ,08,3E,00,3C,42,02,0C,30,40, 794    <093>
1090 DATA 7E,00,3C,42,02,1C,02,42,3C,00,04
     ,0C,14,24,7E,04,04,00,7E,40, 806    <234>
1091 DATA 78,04,02,44,38,00,1C,20,40,7C,42
     ,42,3C,00,7E,42,04,08,10,10, 926    <117>
1092 DATA 10,00,3C,42,42,3C,42,42,3C,00,3C
     ,42,42,3E,02,04,38,00,00,00, 776    <249>
1093 DATA 00,00,FF,00,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,00,00, 510    <112>
1094 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <084>
1095 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <085>
1096 DATA 00,00,00,00,00,00,FF,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 510    <247>
1097 DATA 00,00,FF,00,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,FF,00, 765    <009>
1098 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <088>
1099 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,E0,00, 224    <030>
1100 DATA 00,00,00,00,00,00,FF,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 510    <251>
1101 DATA 00,00,FF,00,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,FF,00, 765    <013>
1102 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <092>
1103 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <093>
1104 DATA 00,00,00,00,00,00,E0,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 479    <242>
1105 DATA 00,00,FF,00,00,00,00,00,00,00,FF
     ,00,00,00,00,00,00,00,FF,00, 765    <017>
1106 DATA 00,00,00,00,00,00,FF,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 255    <071>
1107 DATA 00,00,00,00,00,00,00,00,00,00,00
     ,00,00,00,00,00,00,00,00,00, 0      <097>
1108 DATA 00,00,FC,84,84,84,84,E4,FC,FC,30
     ,50,90,10,10,10,10,7C,7E,82, 2228   <029>
1109 DATA 84,08,10,20,40,FE,7E,82,04,18,18
     ,04,82,7E,18,18,28,48,88,88, 1506   <149>
1110 DATA 7E,08,FC,80,C0,3C,04,04,04,FC,FE
     ,80,80,FE,82,C2,E2,FE,7E,84, 2856   <196>
1111 DATA 04,08,10,20,40,40,FE,84,84,FE,84
     ,84,E2,FE,7E,82,C2,FE,02,02, 2412   <209>
1112 DATA 82,7C,00,00,00,01,01,01,01,01,01
     ,78,FC,C6,83,03,03,03,01,01, 972    <181>
1113 DATA 01,01,FF,01,01,01,03,03,03,03,FF
     ,07,0F,1E,01,01,01,01,FF,07, 845    <029>
1114 DATA 0F,1E,3E,7C,F8,F0,FF,C0,80,80,3C
     ,78,F0,F0,FF,F0,F1,E3,80,80, 3301   <060>
1115 DATA 80,80,FF,80,F0,FC,E7,E6,E6,64,FF
     ,72,38,18,FE,9E,8E,87,FF,83, 3446   <105>
1116 DATA 83,82,0C,07,01,00,FF,00,00,00,82
     ,8E,F8,B0,FF,80,80,80,18,3C, 1955   <036>
1117 DATA 7E,FF,7E,3F,18,00,80,80,80,80,80
     ,80,00,00,0F,1F,1C,30,FF,40, 1803   <144>
1118 DATA C0,C0,83,F2,38,38,FF,1F,0F,0E,F0
     ,FC,FE,FE,FF,FC,78,10,05,07, 2839   <189>
1119 DATA 07,07,FF,0F,0F,0F,0F,1F,1E,1E,FF
     ,3C,38,70,00,00,01,03,FF,0C, 1174   <102>
1120 DATA 38,E0,F0,E0,C0,80,FF,00,00,00,00
     ,00,01,D1,FF,D7,DF,E9,00,00, 2455   <185>
1121 DATA F0,D8,FF,C6,C3,C3,D7,D3,D0,D0,DF
     ,D0,D0,D0,83,01,01,03,FF,07, 3386   <101>
1122 DATA 06,0C,D0,D6,DF,DF,FF,D6,D0,D0,38
     ,E0,C0,00,FF,E0,70,38,D0,D0, 3306   <094>
1123 DATA D0,D0,DF,D0,D3,D7,1C,0E,07,03,FF
     ,03,03,03,DF,D7,D3,D3,FF,D0, 2912   <024>
1124 DATA D0,D0,E6,C6,84,88,FF,00,00,00,00
     ,00,00,00,FF,00,00,00,00,00, 1622   <035>
1125 DATA 00,00,00,00,00,00,10,10,10,10,1F
     ,10,10,10,0B,0B,0B,0B,FB,0B, 449    <252>
1126 DATA 0B,0B,5C,AA,DD,BD,5C,03,8D,B7,1C
     ,BD,5F,03,8D,B8,1C,20,B6,1C, 2023   <180>
1127 DATA BC,5C,03,C0,FF,D0,03,FE,5F,03,C8
     ,48,98,9D,5C,03,BD,5C,03,8D, 2394   <028>
1128 DATA B7,1C,BD,5F,03,8D,B8,1C,68,20,BB
     ,1C,8C,49,03,8D,46,03,A8,B9, 1985   <230>
1129 DATA 97,1D,8D,5A,03,20,B6,1C,20,BB,1C
     ,8C,62,03,8D,63,03,4C,E3,12, 1708   <117>
1130 DATA AD,74,4C,60,EA,48,0A,0A,0A,4A,4A
     ,4A,A8,68,4A,4A,4A,4A,4A,60, 1757   <138>
1131 DATA 04,03,03,03,03,02,02,02,01,01,01
     ,03,02,02,02,02,01,01,01,E8, 271    <071>
1132 DATA 00,A4,AA,AE,B3,B7,BB,BF,C3,C6,C9
     ,CC,CF,D2,D5,D7,D9,DB,DD,DF, 3771   <098>
1133 DATA E1,E3,E4,E6,E7,E8,EA,EB,EC,ED,EE
     ,EF,33,FA,00,3C,66,81,42,81, 3579   <158>
1134 DATA 66,3C,00,3C,7E,FF,FF,FF,7E,3C,07
     ,32,11,31,10,30,0F,0E,2E,0D, 1574   <225>
1135 DATA 2D,0C,0B,2B,0A,2A,09,29,08,07,27
     ,06,26,05,04,24,03,23,02,22, 430    <069>
1136 DATA 01,00,44,FE,44,44,FF,44,FE,44,FF
     ,FF,FF,FF,FF,00,00,00,00,00, 2379   <215>
1137 DATA 00,FF,FF,FF,FF,FF,00,00,00,00,00
     ,00,20,30,18,1C,1E,1F,3F,3F, 1594   <036>
1138 DATA 7F,FE,FC,F8,7C,1E,0E,07,23,FF,FF
     ,FF,FF,FC,FE,7E,3F,0F,03,01, 2825   <125>
1139 DATA 00,00,00,00,00,00,01,01,01,02,39
     ,39,3D,07,03,01,01,01,39,39, 307    <008>
1140 DATA 3D,07,03,01,01,01,39,39,3D,07,03
     ,01,01,01,39,39,3D,07,03,01, 448    <192>
1141 DATA 01,01,FE,C9,4D,BF,5F,2F,17,0B,05
     ,02,FF,08,0A,0B,1B,1D,1E,0D, 1291   <027>
1142 DATA 84,D8,4C,9E,04,F8,CD,8D,1F,D1,1D
     ,1D,10,1E,1E,16,16,37,03,03, 1659   <085>
1143 DATA 03,02,02,02,01,01,01,01,01,01,01
     ,00,50,A8,13,AD,45,03,C9,00, 729    <095>
1144 DATA D0,01,60,AE,44,03,A0,00,B9,30,1D
     ,1D,00,18,9D,00,18,C8,E8,C0, 1830   <127>
1145 DATA 08,D0,F1,60,AD,3F,03,8D,0C,90,8E
     ,3F,03,8E,43,03,60,60,EA,EA, 2169   <214>
1146 DATA EA,EA,EA,EA,B3,23, 1150          <242>
```


Listing 1. Ladeprogramm zu »MUSIC MAESTRO« (Schluß)

```
60 DIM T%(7,3),V%(9),TS%(7):FOR I=0 TO 2:P
   OKE 886+I,0:NEXT:POKE 844,0              <202>
70 POKE 56,76:POKE 55,0:POKE 650,128:POKE
   36879,8:POKE 36878,15:GOSUB 32000        <239>
90 TA(0)=19500:TE(0)=21499:TA(1)=21500:TE(
   1)=23499:TA(2)=23500:TE(2)=24570         <165>
110 FOR J=0 TO 3:FOR I=0 TO 7: READ T%(I,J
   ):NEXT:NEXT                              <175>
120 DATA 0,4,6,0,0,0,0,2,0,16,18,7,9,11,12
   ,14                                      <210>
130 DATA 0,28,30,19,21,23,24,26, 0,0,0,31,
   0,0,0,0                                  <101>
150 TS%(0)=67:TS%(1)=68:TS%(2)=69:TS%(3)=7
   0:TS%(4)=71:TS%(5)=65:TS%(6)=66          <219>
160 CC(1)=176:CC(2)=191:CC(3)=188:CC(4)=17
   2:CC(5)=177:CC(6)=187:CC(7)=165          <188>
170 LF(1)=0:LF(2)=1:LF(3)=0:LF(4)=1:LF(5)=
   0:LF(6)=1:LF(7)=1                        <024>
172 LG(1)=1:LG(2)=0:LG(3)=1:LG(4)=0:LG(5)=
   1:LG(6)=0:LG(7)=1                        <082>
174 VV(1)=44:VV(2)=44:VV(3)=22:VV(4)=22:VV
   (5)=00:VV(6)=00:VV(7)=66                 <242>
176 VX(1)=66:VX(2)=44:VX(3)=44:VX(4)=22:VX
   (5)=22:VX(6)=0:VX(7)=0                   <010>
180 QW(0)=2:QW(1)=1:QW(2)=5:QW(3)=3:QW(4)=
   7:QW(5)=4:QW(6)=6                        <158>
200 POKE 36878,15:POKE 36879,8:PRINT"{CLR,
   WHITE,RVSON,4RIGHT}MUSIC MAESTRO":PRIN
   T"{DOWN,4RIGHT,CYAN}BY C. NEUPERT"       <130>
210 :PRINT"{3DOWN,WHITE,RVSON}1{RVOFF,SPAC
   E,YELLOW}TOENE EINGEBEN/EDIT.":PRINT"{
   WHITE,RVSON}2{RVOFF,SPACE,YELLOW}TOENE
   SPIELEN".                                <211>
220 PRINT"{DOWN,WHITE,RVSON}3{RVOFF,SPACE,
   YELLOW}TOENE RESERVIEREN":PRINT"{DOWN,
   WHITE,RVSON}4{RVOFF,SPACE,YELLOW}TRANS
   PONIEREN"                                <045>
230 PRINT"{DOWN,WHITE,RVSON}5{RVOFF,SPACE,
   YELLOW}SAVE":PRINT"{DOWN,WHITE,RVSON}6
   {RVOFF,SPACE,YELLOW}LOAD"                <090>
240 PRINT"{3DOWN,RED,3RIGHT}WAEHLEN SIE...
   {WHITE}"                                 <225>
250 GET A$:IF A$="" OR VAL(A$)<1 OR VAL(A$
   )> 6 THEN 250                            <112>
260 GOSUB 900: PRINT"{CLR,WHITE}":ON VAL(A
   $)GOTO 1000,10000,5000,6000,7000,8000    <186>
270 STOP                                    <082>
800 PRINT"{DOWN}FUER DIESE STIMME":PRINT"{
   DOWN}WURDEN KEINE TOENE":PRINT"{DOWN}R
   ESERVIERT !":PRINT:RETURN                <013>
900 POKE 36876,240:FOR QR=1 TO 120:NEXT:PO
   KE 36876,0:RETURN                        <251>
1000 POKE 36878,15:PRINT CHR$(147):POKE 36
   879,8                                    <087>
1010 PRINT"{DOWN,WHITE}WELCHE STIMME":PRIN
   T:PRINT"{3SPACE,CYAN}[1...3] ?"          <032>
1012 GET A$:IF A$="" OR VAL(A$)<1 OR VAL(A
   $)>3 THEN 1012                           <169>
1013 GOSUB 900:S=VAL(A$)-1:IF ABS(TA(S)-TE
   (S))>15 THEN 1015                        <126>
1014 GOSUB 800:GOTO 1010                    <036>
1015 PRINT"{CLR,DOWN,CYAN}WOLLEN SIE VORZE
   ICHEN":PRINT"{DOWN}EINGEBEN{SPACE,WHI
   TE}[J/N] ?"                              <208>
1016 GET A$:IF A$="" OR A$<>"J" AND A$<>"N
   "THEN 1016                               <188>
1017 IF A$="N" THEN 1041                    <157>
1018 PRINT"{CLR,DOWN,WHITE}0{CYAN,RIGHT}-
   KEIN VORZEICHEN":PRINT"{DOWN,WHITE}1{
   SPACE,CYAN}- #":PRINT"{DOWN,WHITE}2{C
   YAN,SPACE}- B{YELLOW}"                   <049>
1019 FOR I=0 TO 6:PRINT"{HOME,12DOWN,4RIGH
   T}"CHR$(TS%(I));" ?"                     <241>
1020 GET A$:IF A$="" THEN 1020              <180>
1021 GOSUB 900:Q=VAL(A$):IF Q>2 OR (Q=0 AN
   D A$<>"0") THEN 1020                     <019>
1022 IF Q=2 THEN Q=-1                       <139>
1025 V%(TS%(I)-64)=Q:NEXT I                 <093>
1041 PRINT"{CLR,DOWN,3RIGHT,CYAN}SCHLUESSE
   L:{WHITE}":PRINT:PRINT:PRINT:PRINT"1
   - VIOLINSCHLUESSEL":PRINT                <067>
1042 PRINT"2 - B-SCHLUESSEL":PRINT          <036>
1043 PRINT"3 - BASS-SCHLUESSEL"             <069>
1044 GET A$:IF A$="" OR VAL(A$)>3 OR VQL(A
   $)<1 THEN 1044                           <079>
1045 SF=VAL(A$)-1:POKE 834,SF:POKE 835,0:P
   OKE 36869,254                            <193>
1046 PRINT"{CLR}":POKE 36869,254:GOSUB 180
   0                                        <213>
1050 N=TA(S):SG=36876-S:POKE TE(S),255:GOS
   UB 1610:OK=2:GOSUB 1950                  <198>
1090 TF=0                                   <254>
1100 VQ=0:GET A$:PP=PEEK(653):IF A$="" AND
   PP<> 4 THEN 1100                         <170>
1101 IF PP=4 THEN VA=1:PP=0:POKE 36876,243
   ::POKE 7826,30:POKE 38546,5:POKE 3687
   6,0:GOTO 1100                            <059>
1103 IF TF=1 THEN 1500                      <220>
1104 IF A$<>"" THEN Q=ASC(A$):K=Q-64:IF Q>
   136 AND Q<141 THEN 1550                  <018>
1106 IF Q>132 AND Q<137 THEN OK=Q-133:POKE
   36876,245:GOSUB 1880:GOSUB 1950:GOTO
   1100                                     <195>
1107 IF Q<128 THEN 1111                     <133>
1108 IF PEEK(653)=1 THEN Q=Q-128:K=Q-64:IF
   Q>64 AND Q<72 THEN VQ=1:GOTO 1120        <143>
1109 IF PEEK(653)=2 THEN GOSUB 1900:IF Q>6
   4 AND Q<72 THEN VQ=-1:K=Q-64:GOTO 112
   0                                        <098>
1110 GOTO 1100                              <050>
1111 IF A$="="THEN 10000                    <032>
1112 IF A$="↑"THEN DF=1:GOSUB 2000:GOTO 11
   00                                       <091>
1113 IF A$="←"THEN DF=1:GOSUB 2100:GOTO 11
   00                                       <222>
1114 IF A$="P" THEN L=0:TX=0:GOTO 1130      <007>
1115 IF A$="+" THEN GOSUB 3000:GOTO 1100    <245>
1116 IF A$="-" THEN GOSUB 4000: GOTO 1100   <255>
1117 IF A$="£" THEN POKE 36869,240:GOTO 20
   0                                        <111>
1120 IF Q<65 OR Q> 71 THEN 1150             <149>
1121 IF VA=1 THEN VB=-V%(K):VA=0:GOTO 1125  <018>
1122 VB=0                                   <006>
1125 L=T%(K,OK)+VQ+V%(K)+VB:TX=T%(K,OK)     <015>
1126 IF L<0 OR L>31 THEN POKE 36876,245:::
   ::::::POKE 36876,0:GOTO 1100             <207>
1127 REM                                    <173>
1130 TF=1:POKE 36876,243:POKE 7823,29:POKE
   38543,4:POKE 36876,0:SYS 6067:GOTO 1
   100                                      <143>
1150 IF A$="*"THEN POKE 36876,246:POKE N,2
   55:POKE 36876,0:GOTO 1522                <233>
1160 GOTO 1100                              <100>
1500 R=VAL(A$):IF R<1 OR R>7 THEN POKE 368
   76,245:::::::::POKE 36876,0:GOTO 110
   0                                        <023>
1510 TP=(R-1)*32                            <077>
1520 POKE N,L+TP:GOTO 1530                  <033>
1522 POKE 36876,245:POKE 7825,27:POKE 3854
   5,7:GOSUB 1610:FOR ZX=1 TO 2:GOSUB 90
   0:NEXT                                   <014>
1523 FOR ZX=1 TO 1000:NEXT                  <019>
1524 POKE 7825,138                          <184>
1530 IF N<(TE(S)-1) THEN N=N+1:GOTO 1540    <129>
1535 FOR ZX=1 TO 9:GOSUB 900:NEXT:FOR ZX=1
   TO 1000:NEXT                             <161>
1540 GOSUB 1600:TF=0:POKE 7823,28:POKE 385
   43,3:POKE 7826,138:GOTO 1100             <074>
1550 DF=Q-136:IF DF=1 THEN DF=10:GOSUB 200
   0:GOTO 1100                              <204>
1551 IF DF=2 THEN DF=50:GOSUB 2000:GOTO 11
   00                                       <000>
1552 IF DF=3 THEN DF=10:GOSUB 2100:GOTO 11
   00                                       <034>
1553 DF=50                                  <061>
1555 GOSUB 2100:GOTO 1100                   <091>
1600 GOSUB 1610:IF A$<>"*" THEN GOSUB 1620
   :RETURN                                  <038>
```

Listing 2.»MUSIC MAESTRO« - Konzert auf dem VC 20. Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
1610 PRINT"{HOME,3DOWN,4RIGHT,WHITE,RVSON}
     JJJJJJJ{RVOFF,8LEFT}"(N-TA(S))-VR;TA
     B(16)"{RED,RVSON}JJJJJJJ{RVOFF,8LEFT
     }"TE(S)-(N-VR)                        <200>
1612 RETURN                                <146>
1620 POKE 7750,138:POKE 7762,138           <171>
1630 POKE 838,R-1:POKE 841,TX:SYS 6067:IF
     TX>0 THEN SYS 5640:GOTO 1640          <002>
1635 SYS 5490:RETURN                       <179>
1640 POKE SG,PEEK(7390+L): FOR QT=1 TO 10*
     (2↑(7-R)):NEXT                        <004>
1650 POKE SG,0                             <123>
1699 RETURN                                <233>
1800 POKE 36879,8:PRINT"{CLR,WHITE}":GOSUB
      2900:GOSUB 2400                      <052>
1805 AD=7947:ZZ=22:ON SF+1 GOSUB 2500,2600
     ,2700                                 <135>
1810 QZ=0: FOR I=7680+12+10*22 TO 7680+12+
     19*22 STEP 22                         <014>
1820 POKE I,QZ:POKE I+1,QZ+32:POKE I+2,QZ+
     64:QZ=QZ+1:NEXT I                     <133>
1830 FOR I=6144+8*48 TO 6144+8*58:POKE I,P
     EEK(32768+I-6144):NEXT                <186>
1840 POKE 835,0:SYS 6067                   <114>
1845 FOR I=1 TO 21:POKE 7702+I,137:POKE 77
     90+I,137:NEXT                         <229>
1850 POKE 7689,25:POKE 7690,26:POKE 7747,1
     2:POKE 7748,13:POKE 7759,14:POKE 7760
     ,15                                   <078>
1860 POKE 7839,16:POKE 7840,17:            <209>
1870 POKE 7692,49+S:POKE 38412,3           <248>
1880 POKE 7842,49+OK:POKE 7842+30720,7:POK
     E 36876,0                             <218>
1890 RETURN                                <170>
1900 FOR QR=1 TO 7:IF Q=CC(QR)THEN Q=QR+64
     :RETURN                               <090>
1902 NEXT:RETURN                           <063>
1950 FOR I=7857 TO 8121 STEP 22:POKE I,138
     :NEXT                                 <199>
1955 AD=7857+(3-OK)*66:FOR I=AD TO AD+66 S
     TEP 22                                <200>
1960 POKE I,139:POKE I+30720,2:NEXT        <249>
1970 RETURN                                <250>
2000 IF N+DF>=TE(S)THEN FOR QT=1 TO 9:GOSU
     B 900:NEXT: GOSUB 1610:RETURN         <130>
2010 N=N+DF:VR=0                           <234>
2015 Z=PEEK(N):IF Z=255 THEN FOR QT=1 TO 3
     :GOSUB 900:NEXT:GOSUB 1610:RETURN     <241>
2020 Q1=INT(Z/32):Q2=Z-Q1*32:TX=Q2:L=TX:R=
     Q1+1:GOSUB 1600:RETURN                <044>
2100 IF N-DF<TA(S)THEN FOR QT=1 TO 9:GOSUB
      900:NEXT:GOSUB 1610:RETURN           <221>
2120 N=N-DF:VR=0:GOTO 2015                 <162>
2400 RESTORE: FOR I=1 TO 32: READ X:NEXT   <239>
2420 FOR I=6240 TO 6391:READ X:POKE I,X:NE
     XT                                    <113>
2490 RETURN                                <006>
2500 AD=AD-ZZ:IF AD<7680 OR AD>8185 THEN R
     ETURN                                 <084>
2510 QZ=0:FOR I=AD TO AD+6*ZZ STEP ZZ      <104>
2520 POKE I,106+QZ:POKE I+1,106+QZ+1:QZ=QZ
     +2:NEXT                               <117>
2530 AD=AD+ZZ:GOSUB 2800:RETURN            <219>
2600 IF AD>8185 OR AD<7680 THEN RETURN     <078>
2610 QZ=0: FOR I=AD TO AD+4*ZZ STEP ZZ:POK
     E I,127+QZ:POKE I+1,128+QZ:QZ=QZ+2:NE
     XT                                    <238>
2620 GOSUB 2800:RETURN                     <087>
2700 IF AD<7680 OR AD>8185 THEN RETURN     <178>
2710 POKE AD,120:POKE AD+1,121:POKE AD+ZZ,
     122:POKE AD+ZZ+1,123:POKE AD+2*ZZ+1,1
     24                                    <046>
2720 POKE AD+3*ZZ,125:POKE AD+3*ZZ+1,126:P
     OKE AD+2*ZZ,137:POKE AD+4*ZZ,137      <199>
2725 POKE AD+ 4*ZZ+1,137                   <173>
2730 GOSUB 2800:RETURN                     <197>
2800 IF AD+120>8185 OR AD<7681 THEN RETURN <060>
2810 FOR J=0 TO 4:FOR I=AD+2 TO AD+13-3*(Z
     Z=22):POKE I+ZZ*J,137:NEXT:NEXT       <120>
2820 FOR I=AD-1 TO AD +4*ZZ-1 STEP ZZ:POKE
      I,139:POKE I+14-3*(ZZ=22),140:NEXT   <179>
2825 IF ZZ<>22 THEN RETURN                 <122>
2830 PX=AD+2:FOR I=0 TO 6                  <075>
2840 X=V%(QW(I)):IF X=0 THEN 2880          <178>
2850 IF X=1 THEN C1=18:C2=20:C3=21:GOTO 28
     56                                    <062>
2855 C1=19:C2=22:C3=23                     <070>
2856 IF SF=2 THEN 2875                     <070>
2857 IF SF=1 THEN 2865                     <003>
2858 P=PX+VV(QW(I)):IF LF(QW(I))=1 THEN PO
     KE P,C1:PX=PX+1:IF C1=19 THEN POKE P-
     22,24                                 <008>
2859 IF LF(QW(I))=1 THEN 2880              <155>
2861 POKE P,C2:POKE P+22,C3:PX=PX+1:GOTO 2
     880                                   <250>
2865 P=PX+VX(QW(I)):IF LG(QW(I))=1 THEN PO
     KE P,C1:PX=PX+1:IF C1=19 THEN POKE P-
     22,24                                 <025>
2866 IF LG(QW(I))=1 THEN 2880              <178>
2867 POKE P,C2:POKE P+22,C3:PX=PX+1:GOTO 2
     880                                   <000>
2875 P=PX+VV(QW(I))+22:IF LF(QW(I))=1 THEN
      POKE P,C1:PX=PX+1: IF C1=19 THEN POK
     E P-22,24                             <131>
2876 IF LF(QW(I))=1 THEN 2880              <172>
2877 POKE P,C2:POKE P+22,C3:PX=PX+1        <171>
2880 NEXT I                                <170>
2890 RETURN                                <154>
2900 FOR I=7680 TO 8185:POKE I,138:NEXT:RE
     TURN                                  <235>
3000 IF N>=TE(S)-1 OR N<TA(S) THEN FOR ZX=
     1 TO 9:GOSUB 900:NEXT: RETURN         <156>
3005 POKE 36874,130                        <246>
3010 N1=TE(S)-2                            <152>
3020 POKE N1+1,PEEK(N1)                    <226>
3030 N1=N1-1:IF N1>N THEN 3020             <147>
3040 POKE N1,0:POKE 36874,0:POKE 36876,235
     :FOR ZX=1 TO 500:NEXT: POKE 36876,0:R
     ETURN                                 <089>
4000 IF N<=TA(S) OR N>=TE(S) THEN FOR ZX=1
      TO 9:GOSUB 900:NEXT:RETURN           <178>
4005 POKE 36874,140                        <234>
4010 N1=N                                  <126>
4020 POKE N1-1,PEEK(N1):N1=N1+1            <222>
4030 IF N1<TE(S) THEN 4020                 <088>
4040 POKE 36874,0:N=N-1:POKE 36876,237:FOR
      ZX=1 TO 900:NEXT:POKE 36876,0        <107>
4050 RETURN                                <042>
5000 UB=PEEK(56)*256-PEEK(55):OB=24570:FR=
     OB-UB:AF=UB+1                         <029>
5010 PRINT"{CLR,CYAN,DOWN,RIGHT}FREIE TOEN
     E:";FR-35                             <027>
5020 FOR ZX=0 TO 1:PRINT"{2DOWN,WHITE}WIEV
     IEL TOENE WOLLEN":PRINT"{DOWN}SIE FUE
     R STIMME ";ZX+1                       <064>
5030 PRINT"{DOWN}RESERVIEREN ?":PRINT      <113>
5035 INPUT RT(ZX)                          <174>
5040 IF RT(ZX)+12>FR THEN PRINT"{3DOWN,RED
     }ES SIND NUR{GREEN}";FR-15:PRINT"{DOW
     N,RED}TOENE FREI !{WHITE}":PRINT:GOTO
      5035                                 <073>
5045 FR=FR-RT(ZX)-4:NEXT ZX                <102>
5050 T1=RT(0):T2=RT(1):T3=FR               <102>
5060 TA(0)=AF:TE(0)=AF+T1+1                <015>
5080 TA(1)=AF+T1+2:TE(1)=TA(1)+T2          <128>
5100 TA(2)=AF+T1+T2+2:TE(2)=TA(2)+T3       <135>
5110 GOTO 200                              <220>
6000 INPUT"STIMME";S:IF S<1 OR S> 3 THEN 6
     000                                   <001>
6010 S=S-1:IF ABS(TA(S)-TE(S))<10 THEN GOS
     UB 800:GOTO 6000                      <004>
6020 INPUT"{CLR,DOWN}WIEVIELE HALBTOENE";D
     T:IF DT<1 OR DT>29 THEN 6020          <222>
6030 PRINT"{2DOWN}HOEHER [1] ODER":PRINT"T
     IEFER [2]":PRINT:INPUT SF:IF SF<1 OR
     SF>2 THEN 6030                        <033>
6040 IF SF=2 THEN SF=-1                    <192>
6050 FOR I=TA(S) TO TE(S)-1:T=INT(PEEK(I)/
```

Listing 2. »MUSIC MAESTRO« - Konzert auf dem VC 20 (Fortsetzung)

```
     32):X=PEEK(I)-32*T:IF PEEK(I)=255 THE
     N 6100                                  <012>
6055 IF X;SF*DT>1 AND X+SF*DT<32 THEN X=X+
     SF*DT:GOTO 6090                         <055>
6060 X=X-SF*12:GOTO 6055                     <161>
6090 POKE I,X+32*T:POKE 36876,PEEK(7390+X)
     :NEXT                                   <064>
6100 POKE 36876,0:GOTO 200                   <064>
7000 INPUT"{CLR,DOWN,WHITE}STIMME";S:IF S<
     1 OR S>3 THEN PRINT:GOTO 7000           <041>
7010 S=S-1:IF ABS(TA(S)-TE(S))<15 THEN GOS
     UB 800:GOTO 7000                        <000>
7020 INPUT"{3DOWN}VON TON # ";T1:PRINT:INP
     UT"BIS TON # ";T2                       <211>
7030 T1=T1+TA(S):T2=T2+TA(S):IF T1>=T2 OR(
     T1<TA(S)OR T2>TE(S))THEN PRINT:GOTO 7
     020                                     <192>
7050 FOR I=0 TO 3:POKE I,PEEK(43+I):NEXT:P
     RINT"{CLR,WHITE}PO43,PE(0):PO44,PE(1)
     :PO45,PE(2):PO46,PE(3):RUN"             <003>
7060 AA=INT(T1/256):BB=T1-256*AA-1:CC=INT(
     T2/256):DD=T2-256*CC+1                  <057>
7070 POKE 198,2:POKE 631,19:POKE 632,13:PO
     KE 673,AA:POKE 674,BB:POKE 675,CC:POK
     E 676,DD                                <167>
7080 POKE 44,PEEK(673):POKE 43,PEEK(674):P
     OKE 46,PEEK(675):POKE 45,PEEK(676):SA
     VE"",1,1:END                            <025>
8000 POKE 0,PEEK(45):POKE 1,PEEK(46):PRINT
     "{CLR,WHITE}PO45,PE(0):PO46,PE(1):RUN
     "                                       <241>
8010 POKE 198,2:POKE 631,19:POKE 632,13:LO
     AD"",1,1                                <115>
10000 GOSUB 12000                            <106>
10040 GET A$:IF A$="" THEN 10040             <155>
10050 SYS 5120                               <024>
10060 GET A$:IF A$=""OR A$="S" THEN 10060    <233>
10070 IF A$=" "THEN GOSUB 13000:GOSUB 1230
      0:GOTO 10040                           <254>
10080 GOSUB 13000:GOTO 200                   <099>
12000 GOSUB 13000                            <088>
12005 ZX=0:PRINT"{CLR,WHITE}":POKE 36869,2
      40:FOR I=0 TO 2                        <138>
12006 IF ABS(TA(I)-TE(I))<15 THEN 12040      <148>
12008 PRINT:PRINT"{CYAN}STIMME ";I+1;"{WHI
      TE}J/N ?"                              <142>
12010 GET A$:IF A$="" THEN 12010             <216>
12015 GOSUB 900                              <083>
12020 IF A$="J" THEN POKE 880+I,1:ZX=1:GOT
      O 12040                                <061>
12030 IF A$="N" THEN POKE 880+I,0:GOTO 120
      40                                     <175>
12035 GOTO 12010                             <157>
12040 NEXT I:IF ZX=0 THEN 200                <218>
12045 ZX=0:FOR I=0 TO 2:IF PEEK(880+I)=1 T
      HEN POKE 870+I,0:ZX=1                  <151>
12046 NEXT:IF ZX=0 THEN 200                  <132>
12050 FOR I=0 TO 2: IF PEEK(880+I)=0 THEN
      12090                                  <126>
12060 PRINT"{CLR,DOWN,4RIGHT}STIMME ";I+1;
      " :":PRINT:PRINT                       <048>
12070 PRINT"1 - VIOLINSCHLUESSEL":PRINT      <032>
12072 PRINT"2 - B-SCHLUESSEL":PRINT          <144>
12074 PRINT"3 - BASS-SCHLUESSEL"             <178>
12075 GET A$:IF A$="" OR VAL(A$)<1 OR VAL(
      A$)>3 THEN 12075                       <167>
12080 Q=VAL(A$)-1:GOSUB 900:POKE 883+I,Q     <174>
12090 NEXT I                                 <236>
12100 REM                                    <224>
12300 FOR I=0 TO 2:IF PEEK(880+I)=0 THEN 1
      2330                                   <245>
12310 H=INT(TA(I)/256):L=TA(I)-H*256:POKE
      860+I,L:POKE 863+I,H                   <159>
12330 NEXT I                                 <222>
12340 FOR I=890 TO 916:POKE I,0:NEXT         <178>
12350 POKE 848,1:POKE 849,1:POKE 850,1       <089>
12360 PRINT"{CLR,CYAN,6SPACE,RVSON,SPACE}T
       E M P O ":PRINT"{WHITE,3DOWN,RVSON}
      1{RVOFF,SPACE}LARGO":PRINT             <194>
12361 PRINT"{RVSON}2{RVOFF,SPACE}LARGHETTO
      ":PRINT"{DOWN,RVSON}3{RVOFF,SPACE}AD
      AGIO":PRINT"{DOWN,RVSON}4{RVOFF,SPAC
      E}ANDANTE"                             <196>
12362 PRINT"{DOWN,RVSON}5{RVOFF,SPACE}ALLE
      GRO MODERATO":PRINT"{DOWN,RVSON}6{RV
      OFF,SPACE}ALLEGRO":PRINT"{DOWN,RVSON
      }7{RVOFF,SPACE}PRESTO"                 <142>
12363 GET A$:IF A$="" OR VAL(A$)<1 OR VAL(
      A$)> 7 THEN 12363                      <101>
12364 GOSUB 900:POKE 877,8:POKE 876,35*(8-
      VAL(A$)):PRINT"{CLR}"                  <159>
12400 FOR I=0 TO 2:POKE 835,I: SYS 6067:NE
      XT                                     <145>
12405 POKE 36869,254:GOSUB 2900              <112>
12410 POKE 36864,20:POKE 36865,20:POKE 368
      66,143:POKE 36867,192:POKE 36869,254 <137>
12420 ZZ=15:AD=7711:ON PEEK(883)+1 GOSUB 2
      500,2600,2700:GOTO 12430               <076>
12425 GOSUB 2500                             <021>
12430 ZZ=15:AD=7861:ON PEEK(884)+1 GOSUB 2
      500,2600,2700:GOTO 12440               <070>
12435 GOSUB 2500                             <031>
12440 ZZ=15:AD=8026:ON PEEK(885)+1 GOSUB 2
      500,2600,2700:GOTO 12450               <217>
12445 GOSUB 2500                             <041>
12450 QZ=0: FOR I=7686 TO 8166 STEP 15:POK
      E I,QZ:POKE I+1,QZ+32:POKE I+2,QZ+64 <091>
12460 QZ=QZ+1: NEXT                          <018>
12470 FOR I=7830 TO 7979:POKE I+30720,3:NE
      XT                                     <166>
12480 FOR I=7980 TO 8184:POKE I+30720,2:NE
      XT                                     <165>
12490 FOR I=7705 TO 7709:POKE I+30720,7:NE
      XT:FOR I=7855 TO 7859:POKE I+30720,5
      :NEXT                                  <047>
12495 FOR I=8020 TO 8024:POKE I+30720,4:NE
      XT                                     <042>
12499 RETURN                                 <109>
12999 RETURN                                 <101>
13000 POKE 36864,12:POKE 36865,38:POKE 368
      66,150:POKE 36867,174:POKE 36869,240
      :RETURN                                <113>
30000 DATA 000,251,34,34,34,34,35,0, 000,2
      09,217,85,85,83,211,0                  <151>
30010 DATA 000,247,132,231,133,133,132,0,
      000,189,161,189,33,33,189,0            <134>
30020 DATA 0,244,149,150,149,149,244,0, 0,
      190,8,8,8,8,137,0                      <207>
30030 DATA 36,126,36,36,255,36,126,36, 32,
      56,38,34,255,36,56,0                   <094>
30040 DATA 0,0,72,72,255,72,252,72, 72,252
      ,72,72,255,72,0,0                      <159>
30050 DATA 64,64,64,64,255,112,72,68, 72,7
      2,112,112,255,0,0,0                    <019>
30060 DATA 0,0,0,0,255,32,32,32              <213>
30070 DATA 251,128,128,248,8,8,248,0, 226,
      128,128,128,128,128,146,0              <203>
30075 DATA 254,128,128,252,128,128,254,0     <039>
30080 DATA 60,126,254,255,254,188,128,128,
       126,126,64,126,126,64,64,64           <063>
30090 DATA 128,128,252,132,132,252,4,4       <150>
32000 PRINT"{CLR}":POKE 883,0:POKE 884,1:P
      OKE 885,2:GOSUB 12405:POKE 36878,15  <166>
32005 FOR I=0 TO 2:POKE 835,I:SYS 6067:NEX
      T                                      <192>
32010 X=INT(RND(1)*30)+1:Y=INT(RND(1)*6):P
      OKE 838,Y:POKE 841,X:POKE 835,0:POKE
      834,0:SYS 5640                         <012>
32020 POKE 835,1:POKE 834,1:SYS 5640:POKE
      835,2:POKE 834,2:SYS 5640              <045>
32030 Z=PEEK(7390+X):POKE 36876,Z:POKE 368
      75,Z:POKE 36874,Z                      <044>
32040 FOR ZX=1 TO 22*Y:NEXT                  <120>
32050 GET A$:IF A$=""THEN 32005              <007>
32060 POKE 36876,0:POKE 36875,0:POKE 36874
      ,0:GOSUB 13000:RETURN                  <009>
```


Listing 2. »MUSIC MAESTRO« (Schluß)

Ein 6502-Simulator in Basic

Der Simulator ist eine Hilfe für den Einstieg in die Maschinensprache-Programmierung auf dem VC20/C16. Man kann direkt am Bildschirm verfolgen, was sich bei der Abarbeitung eines Programms in der Zentraleinheit des Computers, im 6502/7501-Prozessor tut.

Die meisten Leser, die sich für Maschinensprache interessieren, werden mit Basic vertraut sein, aber in der Regel noch ein paar Schwierigkeiten mit der Funktion einzelner Maschinenbefehle haben. Wir wollen deshalb im folgenden ein Basic-Programm vorstellen, welches die Maschinenbefehle simuliert. Den Prozessorregistern entsprechen dabei einfache Variablen, und der Stapel wird mit Hilfe eines Arrays simuliert. Die momentanen Werte der Prozessorregister werden am Bildschirm in hexadezimaler Form ausgegeben. Durch die Verwendung von Basic dauert die Ausführung eines Befehls etwa 100000mal länger, als der Prozessor selbst arbeitet. Dies ist aber gerade dazu gut, um in einer Art Einzelschrittverfahren die Funktion eines Programms nachzuvollziehen.

Direkt nach dem Starten fragt das Programm nach der Startadresse, ab der ein Programm bearbeitet werden soll. Das zu testende Maschinenprogramm muß also bereits im Speicher stehen. Man kann sowohl eingebaute ROM-Routinen testen, als auch mit einem Assembler erstellte Programme nachvollziehen.

Auch ist es möglich, kleine Programme mit POKE-Befehlen einzugeben und dann ab der entsprechenden Speicherzelle den Test zu starten.

Im Anschluß werden die Werte der Prozessorregister, die an das Programm übergeben werden sollen, erfragt. Dabei kann man den Wert des Prozessorstatusregisters in binärer Form eingeben. Schließlich werden noch bis zu 10 Adressen erfragt, bei denen das Programm anhalten soll. Wird im Programmablauf eine dieser Adressen erreicht, so hält das Programm an. Alle Eingaben – außer den Daten des Prozessorstatusregisters – müssen in hexadezimaler Form erfolgen.

So funktioniert der 6502-Simulator

Nun wird das Programm schrittweise abgearbeitet und der aktuelle Stand der Prozessorregister nebeneinander auf dem Bildschirm angezeigt. Durch Drücken der SHIFT-Taste kann man das Programm kurzzeitig anhalten.

Das Programm ist wie folgt aufgebaut:

Die Zeilen 10 bis 90 geben die Kopfzeile des Programms aus und beinhalten die Eingaben. Dort werden auch einige Felder und Funktionen definiert, die zu späteren Berechnungen notwendig sind. In Zeile 70 befindet sich der Aufruf eines Unterprogramms ab Zeile 49000, welches ein Demonstrationsprogramm in die Speicherzellen ab 34768=$8700 »POKEd«. In den Zeilen 100 bis 900 sind einige Unterprogramme enthalten, die der Umrechnung von Hexadezimalzahlen dienen, die hier nicht weiter erläutert werden. Ab Zeile 1000 bis 1060 befindet sich die Schleife, die für jeden Befehl durchlaufen wird. Dabei wird zunächst der aktuelle Stand der Prozessorregister angezeigt (Unterprogrammaufruf 2000), dann die Variable OP mit dem Wert der momentan über den Programmzähler adressierten Speicherzelle besetzt und anschließend das Unterprogramm ab Zeile 1100 aufgerufen, welches einen Befehl auswertet. Schließlich wird noch der Programmzähler um eins erhöht und wieder zur Zeile 1000 gesprungen, wenn kein Abbruchkriterium erfüllt ist.

Das Unterprogramm ab Zeile 1100 ist ein Sprungverteiler auf die einzelnen Programmstücke, jeweils ein Sprung für einen Befehl. Ein unzulässiger Befehl bedeutet immer einen Sprung auf das Unterprogramm ab Zeile 1400, welches mit einer entsprechenden Meldung am Bildschirm endet.

Das Unterprogramm ab Zeile 2000 dient zur Anzeige aller Prozessorregister. Das Statusregister wird dabei binär dargestellt, wobei ein Zeichen durch die Funktion CHR$ (48+N) gebildet wird, was hier einer »0« oder »1« entspricht, je nachdem, ob N gleich 0 oder 1 ist. Die anderen Prozessorregister werden mit Hilfe des Unterprogramms ab Zeile 200 in hexadezimale Zahlen umgewandelt. Diese werden gesammelt in Zeile 2090 in der zweiten Bildschirmzeile ausgegeben.

Die Zeilen 3000 bis 9000 enthalten Unterprogramme, die für die Ausführung von Befehlen notwendig sind, beziehungsweise allgemeine Unterprogramme, die die Programmierung der anderen vereinfachen.

Ab Zeile 3000 wird ein 1-Byte-Operand geholt, was dadurch geschieht, daß der Programmzähler um 1 erhöht wird und der Variablen 0 der Wert der angesprochenen Speicherzelle zugewiesen wird. 0 soll hier die Abkürzung für 1 Byte-Operand sein. Im Unterprogramm ab 4000 wird entsprechend ein 2-Byte-Operand geholt, der in der Variablen 00 abgelegt wird.

Die Unterprogramme ab den Zeilen 5000 bis 6000 fassen die einzelnen Statusflags zur Variablen P zusammen beziehungsweise zerlegen diese.

Ab Zeile 7000 steht das Unterprogramm für die Ausführung eines relativen Sprunges.

Komplizierte Addition

Da die Addition ein komplizierter Prozeß ist, da insbesondere Binär- oder Dezimalarithmetik verwendet werden kann, wird diese in dem Unterprogramm ab Zeile 8000 behandelt. Analog wird ab 8200 die Subtraktion behandelt. In diesen Unterprogrammen wird zunächst erfragt, ob das Dezimalflag gesetzt ist. Wenn nicht, ergibt sich der neue Wert des Akkumulators einfach aus:

Alter Wert + Speicherwert + Carry-Flag

Das neue Carry-Flag wird dann gesetzt, wenn der Wert des Akkumulators größer als 255 ist. Dann wird der Akkumulator noch auf einen Wert bis 255 zurückgesetzt (weil die Variable A, anders als der Akkumulator, Zahlen größer als 255 aufnehmen kann) und das Negativ- und das Zero-Flag, also hier die Variablen N und Z, entsprechend besetzt. Wenn jedoch das Dezimal-Flag gesetzt ist, so werden zuerst die niederwertigen 4 Bit des Akkumulators mit den niederwertigen 4 Bit des Speichers verknüpft und anschließend die höherwertigen 4 Bit. Ein Übertrag kommt immer dann zustande, wenn das Ergebnis größer als 9 ist.

Ab Zeile 10000 stehen die Unterprogrammstücke, die jeweils einem Maschinenbefehl entsprechen. Die Zeilennummern der Unterprogramme sind dabei wie folgt aufgebaut:

Zeilennummer Befehl = 10000 + 100 x Wert Maschinencode

00	BRK	33	illegal	66	ROR Op
01	ORA (Op,X)	34	illegal	67	illegal
02	illegal	35	AND Op,X	68	PLA
03	illegal	36	ROL Op,X	69	ADC #Op
04	illegal	37	illegal	6A	ROR #Op
05	ORA Op	38	SEC	6B	illegal
06	ASL Op	39	AND Op,Y	6C	JMP (Op)
07	illegal	3A	illegal	6D	ADC Op
08	PHP	3B	illegal	6E	ROR Op
09	ORA #Op	3C	illegal	6F	illegal
0A	ASL A	3D	AND Op,X	70	BVS Op
0B	illegal	3E	ROL Op,X	71	ADC (Op),Y
0C	illegal	3F	illegal	72	illegal
0D	ORA Op	40	RTI	73	illegal
0E	ASL Op	41	EOR (Op,X)	74	illegal
0F	illegal	42	illegal	75	ADC Op,X
10	BPL Op	43	illegal	76	ROR Op,X
11	ORA (Op),Y	44	illegal	77	illegal
12	illegal	45	EOR Op	78	SEI
13	illegal	46	LSR Op	79	ADC Op,Y
14	illegal	47	illegal	7A	illegal
15	ORA Op,X	48	PHA	7B	illegal
16	ASL Op,X	49	EOR #Op	7C	illegal
17	illegal	4A	LSR A	7D	ADC Op,X
18	CLC	4B	illegal	7E	ROR Op,X
19	ORA Op,Y	4C	JMP Op	7F	illegal
1A	illegal	4D	EOR Op	80	illegal
1B	illegal	4E	LSR Op	81	STA (Op,X)
1C	illegal	4F	illegal	82	illegal
1D	ORA Op,X	50	BVC Op	83	illegal
1E	ASL Op,X	51	EOR (Op),Y	84	STY Op
1F	illegal	52	illegal	85	STA Op
20	JSR Op	53	illegal	86	STX Op
21	AND (Op,X)	54	illegal	87	illegal
22	illegal	55	EOR Op,X	88	DEY
23	illegal	56	LSR Op,X	89	illegal
24	BIT Op	57	illegal	8A	TXA
25	AND Op	58	CLI	8B	illegal
26	ROL Op	59	EOR Op,Y	8C	STY Op
27	illegal	5A	illegal	8D	STA Op
28	PLP	5B	illegal	8E	STX Op
29	AND #Op	5C	illegal	8F	illegal
2A	ROL #Op	5D	EOR Op,X	90	BCC Op
2B	illegal	5E	LSR Op,X	91	STA (Op),Y
2C	BIT Op	5F	illegal	92	illegal
2D	AND Op	60	RTS	93	illegal
2E	ROL Op	61	ADC (Op,X)	94	STY Op,X
2F	illegal	62	illegal	95	STA Op,X
30	BMI Op	63	illegal	96	STX Op,Y
31	AND (Op),Y	64	illegal	97	illegal
32	illegal	65	ADC Op	98	TYA

Illegale Opcodes

Da wir an dieser Stelle davon ausgehen, daß Basic hinlänglich bekannt ist, werden für die Unterprogramme zu den einzelnen Befehlen keine Erklärungen mehr gegeben. Vielmehr sollen gerade die Unterprogramme in Basic dazu dienen, Ihnen die Arbeitsweise der Befehle zu verdeutlichen.

Wie Sie aus Tabelle 1 ersehen können, sind nicht alle der 256 theoretisch möglichen Operations-Codes mit tatsächlichen Funktionen belegt. Sie können jedoch dem Prozessor einen solchen Operations-Code anbieten. Die Reaktion des Prozessors auf einen solchen Befehl ist durch seine interne Struktur definiert. Bei einigen illegalen Befehlen hört der Prozessor auf zu arbeiten und kann dann nur noch mit einem Reset-Impuls reaktiviert werden. Einige Befehle führen jedoch nicht zum Absturz des Computers, sondern führen exotische Funktionen aus, die eine Kombination von offiziellen Operationen darstellen. So zum Beispiel die folgenden drei Befehle:

A7 aa Der Akkumulator und das X-Register werden mit dem Inhalt der Zero-Page-Adresse aa geladen.

87 aa Das Ergebnis einer UND-Funktion zwischen Akku und X-Register wird in die Zero-Page-Adresse aa gespeichert.

97 aa Das Ergebnis einer UND-Funktion zwischen Akku und X-Register wird in die Zero-Page-Adresse aa+Y gespeichert.

Um das Programm nicht noch weiter aufzublähen, wurde beim 6502/7501-Simulator auf die Auswertung illegaler Opcodes verzichtet. (ev)

Buchauszug »Das Commodore 64 Buch«, Band 4, M&T-Verlag AG

99	STA Op,Y	BB	illegal	DD	CMP Op,X
9A	TXS	BC	LDY Op,X	DE	DEC Op,X
9B	illegal	BD	LDA Op,Y	DF	illegal
9C	illegal	BE	LDX Op,Y	E0	CPX #Op
9D	STA Op,X	BF	illegal	E1	SBC (Op,X)
9E	illegal	C0	CPY #Op	E2	illegal
9F	illegal	C1	CMP (Op,X)	E3	illegal
A0	LDY #Op	C2	illegal	E4	CPX Op
A1	LDA (Op,X)	C3	illegal	E5	SBC Op
A2	LDX #Op	C4	CPY Op	E6	INC Op
A3	illegal	C5	CMP Op	E7	illegal
A4	LDY Op	C6	DEC Op	E8	INX
A5	LDA Op	C7	illegal	E9	SBC #Op
A6	LDX Op	C8	INY	EA	NOP
A7	illegal	C9	CMP #Op	EB	illegal
A8	TAY	CA	DEX	EC	CPX Op
A9	LDA #Op	CB	illegal	ED	SBC Op
AA	TAX	CC	CPY Op	EE	INC Op
AB	illegal	CD	CMP Op	EF	illegal
AC	LDY Op	CE	DEC Op	F0	BEQ Op
AD	LDA Op	CF	illegal	F1	SBC Op,Y
AE	LDX Op	D0	BNE Op	F2	illegal
AF	illegal	D1	CMP (Op),Y	F3	illegal
B0	BCS Op	D2	illegal	F4	illegal
B1	LDA (Op),Y	D3	illegal	F5	SBC Op,X
B2	illegal	D4	illegal	F6	INC Op,X
B3	illegal	D5	CMP Op,X	F7	illegal
B4	illegal	D6	DEC Op,X	F8	SED
B5	LDA Op,X	D7	illegal	F9	SBC Op,Y
B6	LDX Op,Y	D8	CLD	FA	illegal
B7	illegal	D9	CMP Op,X	FB	illegal
B8	CLV	DA	illegal	FC	illegal
B9	LDA Op,Y	DB	illegal	FD	SBC Op,X
BA	TSX	DC	illegal	FE	INC Op,X
				FF	illegal

Tabelle 1. Die Maschinenbefehle des 6502/7501-Prozessors



DEMO6502		10 - 50000	
Variablen:			
Name	Typ	Bereich	Bedeutung
A	G	0...255	Akkumulator
B	G	0,1	Break-Flag
C	G	0,1	Carry-Flag
D	G	0,1	Dezimal-Flag
H	H	Integer	Hilfsvariable
H$	H	2 Zeichen	Hilfsvariable
HO$	H	31 Zeichen	Eingabezeile
HE$	G	"0123456789ABCDEF"	Hexadezimalziffern
HH	H	0...65535	Hilfsvariable
HH$	H	4 Zeichen	Hilfsvariable
HI	H/R	0...255	Höherwertiges Byte
HP	H	0...10	Anzahl Abbruchpunkte
I	G	0,1	Interrupt-Disable-Flag
J	H	Integer	Laufvariable
LO	H/R	0...255	Niederwertiges Byte
M	H	0...65535	Speicheradresse
N	G	0,1	Negativ-Flag
O	H	0...255	Operand (1 Byte)
OO	H	0...65535	Operand (2 Byte)
OP	H	0...255	Befehlscode
P	G	0...255	Prozessorstatus
P5	G	0,1	Bit 5 von P
PC	G	0...65535	Programmzähler
S	G	0...255	Stapelzeiger
V	G	0,1	Overflow-Flag
X	G	0...255	X-Register
Y	G	0...255	Y-Register
Z	G	0,1	Zero-Flag

Felder (Arrays):

Name	Dimen.	Typ	Bereich	Bedeutung
HP	10	G	0...65535	Abbruchstellen
ST%	255	G	0...255	Stack

Tabelle 2a. Variablenliste zu »Demo 6502«

Unterprogrammaufrufe : (nur im Bereich 10 - 8440)

in	nach	Zweck
22	2000	Anzeige Register
26	400	Wert von 4stelliger Hexzahl bestimmen
44	350	Wert von 2stelliger Hexzahl bestimmen
46	350	Wert von 2stelliger Hexzahl bestimmen
48	350	Wert von 2stelliger Hexzahl bestimmen
50	350	Wert von 2stelliger Hexzahl bestimmen
62	400	Wert von 4stelliger Hexzahl bestimmen
70	49000	Beispielprogramm »poken«
270	200	2stellige Hexzahl bilden
300	200	2stellige Hexzahl bilden
420	350	Wert von 2stelliger Hexzahl bestimmen
450	350	Wert von 2stelliger Hexzahl bestimmen
1010	2000	Anzeige Register
1045	200	2-stellige Hexzahl bilden
1050	1100	Befehl ausführen
2010	250	4stellige Hexzahl bilden
2040	200	2stellige Hexzahl bilden
2050	200	2stellige Hexzahl bilden
2060	200	2stellige Hexzahl bilden
2070	200	2stellige Hexzahl bilden

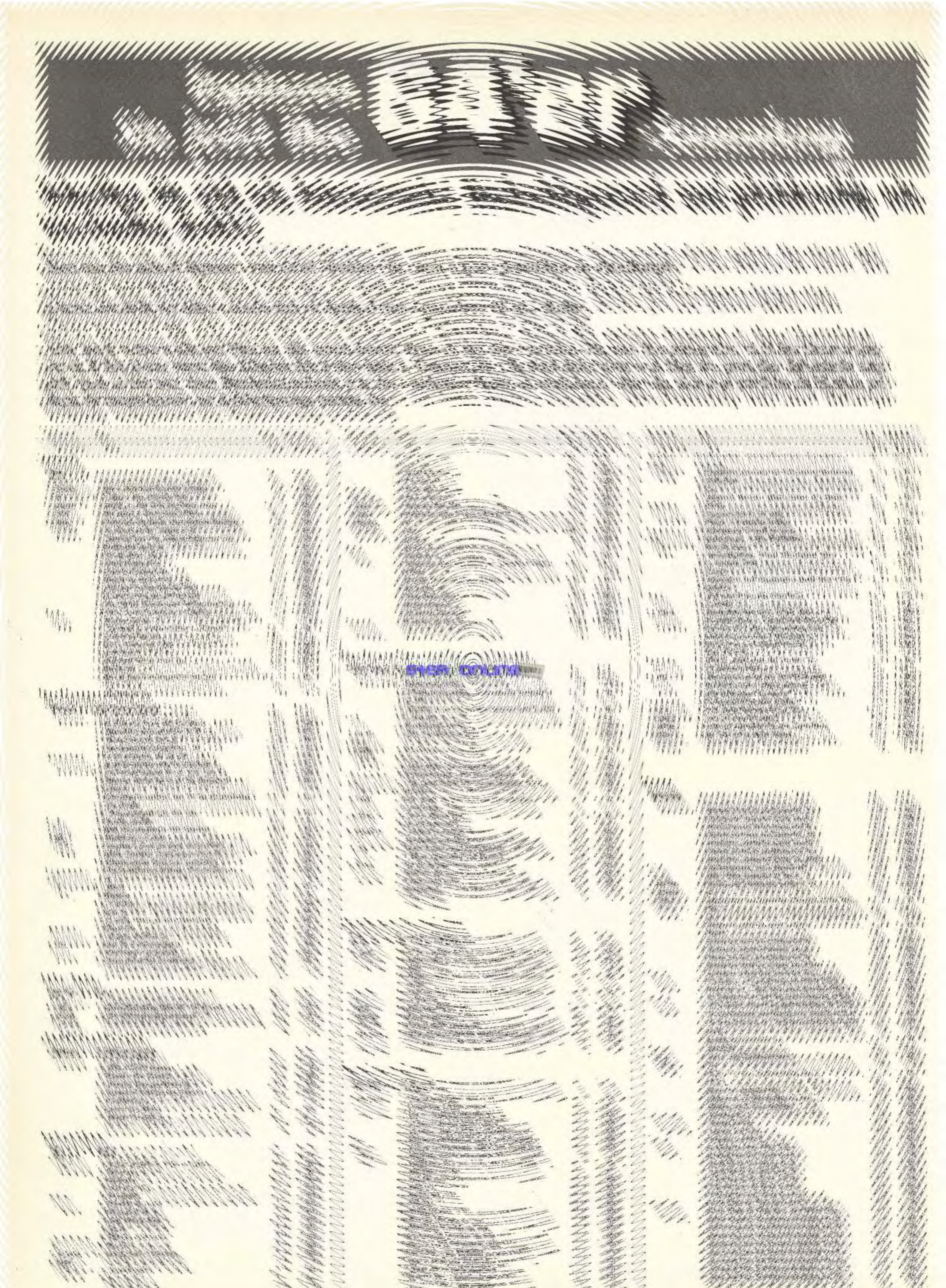
Verzweigungen nach außen:

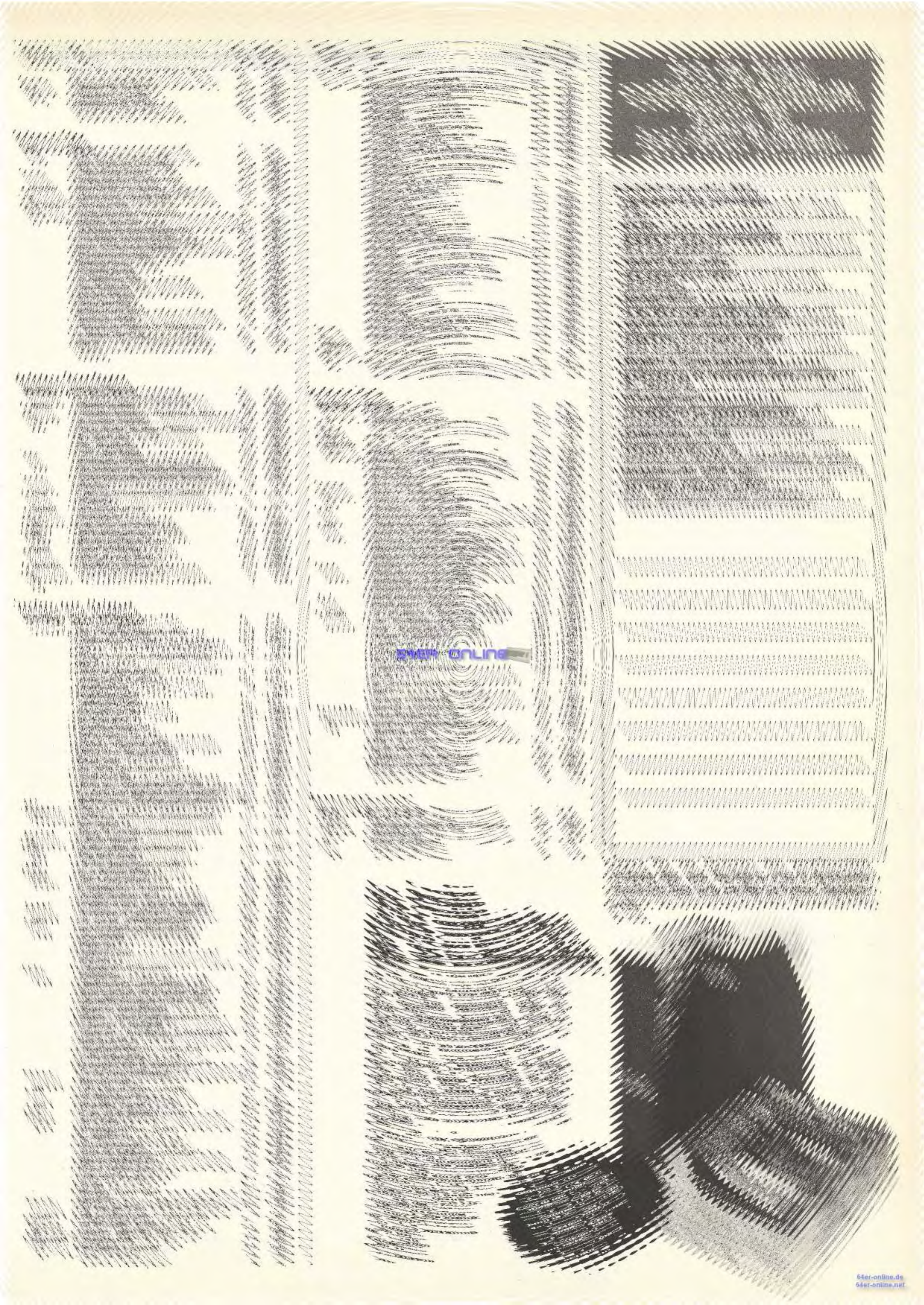
In Ze	nach	Bedingung	Bemerkung
1090	STOP	PC = HP(J)	Abbruchpunkt erreicht
1370	STOP	OP größer als 255	
1410	STOP		illegaler Befehl

Tabelle 2b. Unterprogrammaufruf zu »Demo 6502«

```
10 PRINT "{CLR}{3SPACE}PC{3SPA
   CE}NV-BDIZC{3SPACE}AC{2SPAC
   E}XR{2SPACE}YR{2SPACE}SP     <142>
12 HE$="0123456789ABCDEF"       <255>
14 DIM ST%(255):REM STACK       <210>
16 DEF FN EO(M)=(M AND (NOT A)
   ) OR ((NOT M) AND A)         <182>
18 DEF FN V(X)=X-48+7*(X>64)    <016>
20 DEF FN F0(X)=(ASC(MID$(H0$,
   X,1))-48) AND 1              <180>
22 GOSUB 2000                   <242>
24 INPUT "{HOME}{DOWN}";H0$     <239>
26 HH$=LEFT$(H0$,4):GOSUB 400:
   PC=HH                        <090>
28 N=FN  F0(7)                  <181>
30 V=FN  F0(8)                  <216>
32 P5=FN F0(9)                  <214>
34 B=FN F0(10)                  <055>
36 D=FN F0(11)                  <067>
38 I=FN F0(12)                  <091>
40 Z=FN F0(13)                  <163>
42 C=FN F0(14)                  <075>
44 H$=MID$(H0$,18,2):GOSUB 350
   :A=H                         <254>
46 H$=MID$(H0$,22,2):GOSUB 350
   :X=H                         <110>
48 H$=MID$(H0$,26,2):GOSUB 350
   :Y=H                         <240>
50 H$=MID$(H0$,30,2):GOSUB 350
   :S=H                         <024>
52 PRINT "{2DOWN}"              <102>
54 HP=HP+1                      <167>
56 PRINT HP"{LEFT}. ABBRUCHPUN
   KT";                         <071>
58 INPUT "{3SPACE}{3LEFT}";HH$ <114>
60 IF HH$="X" THEN HP=HP-1:GOT
   O 68                         <045>
62 GOSUB 400                    <030>
64 HP(HP)=HH                    <121>
66 GOTO 54                      <084>
68 PRINT "{UP}{23SPACE}";        <230>
70 GOSUB 49000                  <218>
72 GOTO 1000                    <012>
140 :                           <116>
150 REM *** WERT IN HIGH- UND
    LOW ZERLEGEN ***            <029>
160 HI=INT(HH/256)              <182>
170 LO=HH-256*HI                <246>
180 RETURN                      <238>
190 :                           <166>
200 REM *   2-STELLIGE HEXZAHL
     (H$) AUS H BILDEN
    *                           <187>
210 H$=MID$(HE$,H/16+1,1)+MID$
    (HE$,(H AND 15)+1,1)        <166>
220 RETURN                      <022>
240 :                           <216>
250 REM *   4-STELLIGE HEXZAHL
     (HH$) AUS HH BILDEN
    *                           <196>
260 H=INT(HH/256)               <122>
270 GOSUB 200                   <224>
280 HH$=H$                      <057>
290 H=HH-256*H                  <255>
300 GOSUB 200                   <254>
310 HH$=HH$+H$                  <036>
320 RETURN                      <124>
340 :                           <062>
350 REM * WERT VON 2-STELLIGER
     HEX-ZAHL (H$) -> H
    *                           <084>
360 H=16*FN V(ASC(H$))+FN V(AS
    C(MID$(H$,2)))              <118>
370 RETURN                      <174>
390 :                           <112>
400 REM * WERT VON 4-STELLIGER
     HEX-ZAHL (HH$) -> HH
    *                           <098>
410 H$=LEFT$(HH$,2)             <179>
420 GOSUB 350                   <206>
430 HH=256*H                    <040>
440 H$=RIGHT$(HH$,2)            <241>
450 GOSUB 350                   <236>
460 HH=HH+H                     <020>
470 RETURN                      <018>
990 :                           <204>
1000 REM *** SCHLEIFE FUER JED
     EN BEFEHL ***              <011>
1010 GOSUB 2000                 <212>
1020 IF HP=0 THEN 1000          <126>
1030 FOR J=1 TO HP:IF PC=HP(J)
      THEN 1080                 <024>
1035 NEXT J                     <111>
1040 OP=PEEK(PC)                <033>
1045 H=OP:GOSUB 200:PRINT "OPC
     ODE: "H$"{2SPACE}";        <153>
1050 GOSUB 1100                 <006>
1055 PC=PC+1                    <013>
1060 GOTO 1000                  <240>
1080 PRINT "ABBRUCH (";J;") "; <150>
1090 STOP                       <140>
1100 IF OP=0 THEN 10000         <011>
1110 ON OP    GOTO 10100, 1400
     , 1400, 1400,10500,10600,
```

Listing »Demo 6502«





```
        1400,10800,10900,11000   <136>
1120 ON OP-10 GOTO  1400, 1400
       ,11300,11400, 1400,11600,
       11700, 1400, 1400, 1400   <021>
1130 ON OP-20 GOTO 12100,12200
       , 1400,12400,12500, 1400,
       1400, 1400,12900,13000    <128>
1140 ON OP-30 GOTO  1400,13200
       ,13300, 1400, 1400,13600,
       13700,13800, 1400,14000   <091>
1150 ON OP-40 GOTO 14100,14200
       , 1400,14400,14500,14600,
       1400,14800,14900, 1400    <082>
1160 ON OP-50 GOTO  1400, 1400
       ,15300,15400, 1400,15600,
       15700, 1400, 1400, 1400   <046>
1170 ON OP-60 GOTO 16100,16200
       , 1400,16400,16500, 1400,
       1400, 1400,16900,17000    <038>
1180 ON OP-70 GOTO  1400,17200
       ,17300,17400, 1400,17600,
       17700,17800, 1400,18000   <088>
1190 ON OP-80 GOTO 18100, 1400
       , 1400, 1400,18500,18600,
       1400,18800,18900, 1400    <246>
1200 ON OP-90 GOTO  1400, 1400
       ,19300,19400, 1400,19600,
       19700, 1400, 1400, 1400   <072>
1210 ON OP-100GOTO 20100,20200
       , 1400,20400,20500,20600,
       1400,20800,20900,21000    <237>
1220 ON OP-110GOTO  1400,21200
       ,21300, 1400, 1400, 1400,
       21700,21800, 1400,22000   <018>
1230 ON OP-120GOTO 22100, 1400
       , 1400, 1400,22500,22600,
       1400, 1400,22900, 1400    <053>
1240 ON OP-130GOTO  1400,23200
       ,23300,23400, 1400,23600,
       1400,23800, 1400,24000    <066>
1250 ON OP-140GOTO 24100,24200
       , 1400,24400,24500, 1400,
       1400,24800,24900,25000    <089>
1260 ON OP-150GOTO  1400,25200
       ,25300,25400, 1400, 1400,
       25700, 1400, 1400,26000   <165>
1270 ON OP-160GOTO 26100,26200
       , 1400,26400,26500,26600,
       1400,26800,26900,27000    <037>
1280 ON OP-170GOTO  1400,27200
       ,27300,27400, 1400,27600,
       27700, 1400, 1400,28000   <055>
1290 ON OP-180GOTO 28100,28200
       , 1400,28400,28500,28600,
       1400,28800,28900,29000    <143>
1300 ON OP-190GOTO  1400,29200
       ,29300, 1400, 1400,29600,
       29700,29800, 1400,30000   <242>
1310 ON OP-200GOTO 30100,30200
       , 1400,30400,30500,30600,
       1400,30800,30900, 1400    <169>
1320 ON OP-210GOTO  1400, 1400
       ,31300,31400, 1400,31600,
       31700, 1400, 1400, 1400   <104>
1330 ON OP-220GOTO 32100,32200
       , 1400,32400,32500, 1400,
       1400,32800,32900,33000    <212>
1340 ON OP-230GOTO  1400,33200
       ,33300,33400, 1400,33600,
       33700,33800, 1400,34000   <101>
1350 ON OP-240GOTO 34100, 1400
       , 1400, 1400,34500,34600,
       1400,34800,34900, 1400    <224>
1360 ON OP-250GOTO  1400, 1400
       ,35300,35400, 1400        <194>
1370 STOP                        <166>
1400 PRINT "UNGUELTIGER CODE";   <011>
1410 STOP                        <206>
1990 :                           <188>
2000 REM *** ANZEIGE ***         <070>
2010 HH=PC:GOSUB 250:PC$=HH$     <198>
2020 ST$=CHR$(48+N)+CHR$(48+V)
     +CHR$(48+P5)+CHR$(48+B)     <249>
2030 ST$=ST$+CHR$(48+D)+CHR$(4
     8+I)+CHR$(48+Z)+CHR$(48+C
     )                           <129>
2040 H=A:GOSUB 200:A$=H$         <209>
2050 H=X:GOSUB 200:X$=H$         <006>
2060 H=Y:GOSUB 200:Y$=H$         <040>
2070 H=S:GOSUB 200:S$=H$         <162>
2090 PRINT "{HOME}{DOWN}{2SPAC
     E}"PC$"{2SPACE}"ST$"{3SPA
     CE}"A$"{2SPACE}"X$"{2SPAC
     E}"Y$"{2SPACE}"S$           <071>
2100 RETURN                      <126>
2990 :                           <172>
3000 REM *** 1-BYTE-OPERAND HO
     LEN ***                     <242>
3010 PC=PC+1                     <190>
3020 O=PEEK(PC)                  <195>
3030 RETURN                      <038>
3990 :                           <156>
4000 REM *** 2-BYTE-OPERAND HO
     LEN ***                     <034>
4010 PC=PC+1                     <174>
4020 O=PEEK(PC)                  <179>
4030 PC=PC+1                     <194>
4040 OO=O+256*PEEK(PC)           <205>
4050 RETURN                      <042>
4990 :                           <140>
5000 REM *** STATUS ZUSAMMENFA
     SSEN ***                    <176>
5010 P=128*N+64*V+32*P5+16*B+8
     *D+4*I+2*Z+C                <211>
5020 RETURN                      <252>
5990 :                           <124>
6000 REM *** STATUS ZERLEGEN *
     **                          <188>
6010 N =(P AND 128)/128          <159>
6020 V =(P AND  64)/ 64          <098>
6030 P5=(P AND  32)/ 32          <226>
6040 B =(P AND  16)/ 16          <157>
6050 D =(P AND   8)/  8          <109>
6060 I =(P AND   4)/  4          <121>
6070 Z =(P AND   2)/  2          <190>
6080 C =P AND 1                  <214>
6090 RETURN                      <050>
6990 :                           <108>
7000 REM *** RELATIVEN SPRUNG
     AUSFUEHREN ***              <123>
7010 IF O>127 THEN O=O-256       <153>
7020 PC=PC+O                     <166>
7030 RETURN                      <230>
7990 :                           <092>
8000 REM *** A=A+M+C (ADC-BEFE
     HL) ***                     <060>
8010 IF D=0 THEN A=A+M+C:C=-(A
     >255):GOTO 8070             <143>
8020 H=(A AND 15)+(M AND 15)+C   <033>
8030 C=-(H>9)                    <114>
8040 H=H AND 15                  <187>
8050 A=(A AND 240)+(M AND 240)
     +H+16*C                     <179>
8060 C=-(A>99)                   <146>
8070 A=A AND 255                 <194>
8080 N=(A AND 128)/128           <227>
8090 Z=-(A=0)                    <192>
8100 RETURN                      <028>
8190 :                           <036>
8200 REM *** A=A-M-1+C (SBC-BE
     FEHL) ***                   <002>
8210 IF D=0 THEN A=A-M-1+C:C=-
     (A>=0):GOTO 8270            <044>
8220 H=(A AND 15)-(M AND 15)-1
     +C                          <002>
8230 C=-(H>=0)                   <165>
8240 H=H AND 15                  <057>
8250 A=(A AND 240)-(M AND 240)
     +H-16+16*C                  <239>
8260 C=-(A>=0)                   <226>
8270 A=A AND 255                 <203>
8280 C=(A AND 128)/128           <074>
8290 Z=-(A=0)                    <212>
8300 RETURN                      <137>
8390 :                           <058>
8400 REM *** A-M (CMP-BEFEHL)
     ***                         <086>
8410 C=-((A-M)>=0)               <085>
8420 N=((A-M) AND 128)/128       <239>
8430 Z=-((A-M)=0)                <241>
8440 RETURN                      <021>
9990 :                           <060>
10000 REM *** BRK                <216>
10005 HH=PC+2:GOSUB 150          <222>
10010 ST%(S)=HI:S=(S-1) AND 25
      5                          <208>
10015 ST%(S)=LO:S=(S-1) AND 25
      5                          <245>
10020 B=1                        <035>
10025 GOSUB 5000                 <107>
10030 ST%(S)=P:S=(S-1) AND 255   <162>
10035 I=1                        <078>
10040 PC=PEEK(65534)+256*PEEK(
      65535)                     <243>
10045 GOSUB 2000                 <103>
10050 PRINT "BRK ERREICHT";      <220>
10055 END                        <151>
10090 :                          <160>
10100 REM *** ORA (INDIRECT,X)   <238>
10110 GOSUB 3000                 <176>
10120 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                       <030>
10130 A=A OR PEEK(OO)            <249>
10140 N=(A AND 128)/128          <255>
10150 Z=-(A=0)                   <220>
10160 RETURN                     <056>
10490 :                          <050>
10500 REM *** ORA ZEROPAGE       <154>
10510 GOSUB 3000                 <068>
10520 A=A OR PEEK(O)             <147>
10530 N=(A AND 128)/128          <137>
10540 Z=-(A=0)                   <102>
10550 RETURN                     <194>
10590 :                          <152>
10600 REM *** ASL ZEROPAGE       <006>
10610 GOSUB 3000                 <168>
10620 M=PEEK(O)                  <048>
10630 M=M*2                      <151>
10640 C=-(M>255)                 <039>
10650 M=M AND 255                <218>
10660 POKE O,M                   <057>
10670 Z=-(M=0)                   <235>
10680 N=(M AND 128)/128          <160>
10690 RETURN                     <078>
10790 :                          <098>
10800 REM *** PHP                <130>
10810 GOSUB 5000                 <130>
10820 ST%(S)=P:S=(S-1) AND 255   <190>
10830 RETURN                     <220>
10890 :                          <198>
10900 REM *** ORA #IMMEDIATE     <159>
10910 GOSUB 3000                 <214>
10920 A=A OR O                   <147>
10930 N=(A AND 128)/128          <027>
10940 Z=-(A=0)                   <248>
10950 RETURN                     <084>
10990 :                          <042>
11000 REM *** ASL A              <137>
11010 A=A*2                      <039>
11020 C=-(A>255)                 <164>
11030 A=A AND 255                <106>
11040 Z=-(A=0)                   <157>
11050 N=(A AND 128)/128          <149>
11060 RETURN                     <196>
11290 :                          <090>
11300 REM *** ORA ABSOLUTE       <149>
11310 GOSUB 4000                 <114>
11320 A=A OR PEEK(OO)            <169>
11330 N=(A AND 128)/128          <175>
11340 Z=-(A=0)                   <140>
11350 RETURN                     <232>
11390 :                          <190>
11400 REM *** ASL ABSOLUTE       <001>
11410 GOSUB 4000                 <214>
11420 M=PEEK(OO)                 <146>
11430 M=M*2                      <189>
11440 C=-(M>255)                 <077>
11450 M=M AND 255                <000>
11460 POKE OO,M                  <192>
11470 Z=-(M=0)                   <017>
11480 N=(M AND 128)/128          <198>
11490 RETURN                     <116>
11590 :                          <136>
11600 REM *** BPL                <036>
11610 GOSUB 3000                 <152>
11620 IF N=0 THEN GOSUB 7000     <022>
11630 RETURN                     <002>
11690 :                          <236>
11700 REM ** ORA (INDIRECT),Y    <097>
11710 GOSUB 3000                 <252>
11720 OO=PEEK(O)+256*PEEK(O+1)
      +Y                         <062>
11730 A=A OR PEEK(OO)            <069>
11740 N=(A AND 128)/128          <075>
11750 Z=-(A=0)                   <040>
11760 RETURN                     <132>
12090 :                          <128>
12100 REM *** ORA ZEROPAGE,X     <159>
12110 GOSUB 3000                 <144>
12120 A=A OR PEEK(O+X)           <140>
12130 N=(A AND 128)/128          <213>
12140 Z=-(A=0)                   <178>
12150 RETURN                     <014>
12190 :                          <228>
12200 REM *** ASL ZEROPAGE,X     <011>
12210 GOSUB 3000                 <244>
12220 M=PEEK(O+X)                <231>
12230 M=M*2                      <227>
12240 C=-(M>255)                 <115>
12250 M=M AND 255                <038>
12260 POKE O+X,M                 <128>
12270 Z=-(M=0)                   <055>
12280 N=(M AND 128)/128          <236>
12290 RETURN                     <156>
12390 :                          <174>
12400 REM *** CLC                <127>
12410 C=0                        <127>
12420 RETURN                     <030>
12490 :                          <018>
12500 REM *** ORA ABSOLUTE,Y     <010>
12510 GOSUB 4000                 <042>
12520 A=A OR PEEK(OO+Y)          <195>
12530 N=(A AND 128)/128          <103>
12540 Z=-(A=0)                   <068>
12550 RETURN                     <162>
12890 :                          <166>
12900 REM *** ORA ABSOLUTE,X     <154>
12910 GOSUB 4000                 <190>
12920 A=A OR PEEK(OO+X)          <079>
12930 N=(A AND 128)/128          <251>
12940 Z=-(A=0)                   <216>
12950 RETURN                     <052>
12990 :                          <010>
13000 REM *** ASL ABSOLUTE,X     <006>
13010 GOSUB 4000                 <034>
13020 M=PEEK(OO+X)               <179>
13030 M=M*2                      <009>
13040 C=-(M>255)                 <153>
13050 M=M AND 255                <076>
13060 POKE OO+X,M AND 255        <039>
13070 Z=-(M=0)                   <095>
13080 N=(M AND 128)/128          <020>
13090 RETURN                     <194>
13190 :                          <212>
13200 REM *** JSR                <249>
```

```
13210 HH=PC+2:GOSUB 150           <123>
13220 ST%(S)=HI:S=(S-1) AND 25
      5                           <114>
13230 ST%(S)=LO:S=(S-1) AND 25
      5                           <156>
13240 GOSUB 4000                  <010>
13250 PC=OO-1                     <112>
13260 RETURN                      <108>
13290 :                           <056>
13300 REM *** AND (INDIRECT,X)    <004>
13310 GOSUB 3000                  <072>
13320 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                        <184>
13330 A=A AND PEEK(OO)            <114>
13340 N=(A AND 128)/128           <153>
13350 Z=-(A=0)                    <118>
13360 RETURN                      <210>
13590 :                           <104>
13600 REM *** BIT ZEROPAGE        <250>
13610 GOSUB 3000                  <120>
13620 M=PEEK(O)                   <000>
13630 N=(M AND 128)/128           <062>
13640 V=(M AND 64)/64             <001>
13650 Z=-((A AND M)=0)            <205>
13660 RETURN                      <000>
13690 :                           <204>
13700 REM *** AND ZEROPAGE        <176>
13710 GOSUB 3000                  <220>
13720 A=A AND PEEK(O)             <010>
13730 N=(A AND 128)/128           <033>
13740 Z=-(A=0)                    <254>
13750 RETURN                      <090>
13790 :                           <048>
13800 REM *** ROL ZEROPAGE        <224>
13810 GOSUB 3000                  <064>
13820 M=PEEK(O)                   <200>
13830 M=M*2+C                     <201>
13840 C=-(M>255)                  <193>
13850 M=M AND 255                 <116>
13860 POKE O,M AND 255            <118>
13870 Z=-(M=0)                    <133>
13880 N=(M AND 128)/128           <058>
13890 RETURN                      <232>
13990 :                           <250>
14000 REM *** PLP                 <028>
14010 S=(S+1) AND 255             <130>
14020 P=ST%(S)                    <199>
14030 GOSUB 6000                  <054>
14040 RETURN                      <126>
14090 :                           <096>
14100 REM *** AND #IMMEDIATE      <183>
14110 GOSUB 3000                  <112>
14120 A=A AND O                   <012>
14130 N=(A AND 128)/128           <181>
14140 Z=-(A=0)                    <146>
14150 RETURN                      <238>
14190 :                           <196>
14200 REM *** ROL A               <101>
14210 A=A*2+C                     <087>
14220 C=-(A>255)                  <060>
14230 A=A AND 255                 <002>
14240 Z=-(A=0)                    <246>
14250 N=(A AND 128)/128           <045>
14260 RETURN                      <092>
14390 :                           <142>
14400 REM *** BIT ABSOLUTE        <020>
14410 GOSUB 4000                  <166>
14420 M=PEEK(OO)                  <098>
14430 N=(M AND 128)/128           <100>
14440 V=(M AND 64)/64             <039>
14450 Z=-((A AND M)=0)            <243>
14460 RETURN                      <038>
14490 :                           <242>
14500 REM *** AND ABSOLUTE        <171>
14510 GOSUB 4000                  <010>
14520 A=A AND PEEK(OO)            <032>
14530 N=(A AND 128)/128           <071>
14540 Z=-(A=0)                    <036>
14550 RETURN                      <128>
14590 :                           <086>
14600 REM *** ROL ABSOLUTE        <221>
14610 GOSUB 4000                  <112>
14620 M=PEEK(OO)                  <044>
14630 M=M*2+C                     <239>
14640 C=-(M>255)                  <231>
14650 M=M AND 255                 <154>
14660 POKE OO,M AND 255           <161>
14670 Z=-(M=0)                    <171>
14680 N=(M AND 128)/128           <096>
14690 RETURN                      <014>
14790 :                           <032>
14800 REM *** BMI                 <055>
14810 GOSUB 3000                  <048>
14820 IF N=1 THEN GOSUB 7000      <206>
14830 RETURN                      <154>
14890 :                           <134>
14900 REM *** AND (INDIRECT),Y    <232>
14910 GOSUB 3000                  <150>
14920 OO=PEEK(O)+256*PEEK(O+1)
      +Y                          <216>
14930 A=A AND PEEK(OO)            <190>
14940 N=(A AND 128)/128           <229>
14950 Z=-(A=0)                    <194>
14960 RETURN                      <030>
15290 :                           <024>
15300 REM *** AND ZEROPAGE,X      <181>
15310 GOSUB 3000                  <040>
15320 A=A AND PEEK(O+X)           <003>
15330 N=(A AND 128)/128           <109>
15340 Z=-(A=0)                    <074>
15350 RETURN                      <166>
15390 :                           <126>
15400 REM ***ROL ZEROPAGE,X       <231>
15410 GOSUB 3000                  <142>
15420 M=PEEK(O+X)                 <129>
15430 M=M*2+C                     <021>
15440 C=-(M>255)                  <013>
15450 M=M AND 255                 <192>
15460 POKE O+X,M AND 255          <170>
15470 Z=-(M=0)                    <209>
15480 N=(M AND 128)/128           <134>
15490 RETURN                      <052>
15590 :                           <070>
15600 REM *** SEC                 <151>
15610 C=1                         <039>
15620 RETURN                      <184>
15690 :                           <172>
15700 REM *** AND ABSOLUTE,Y      <034>
15710 GOSUB 4000                  <196>
15720 A=A AND PEEK(OO+Y)          <060>
15730 N=(A AND 128)/128           <001>
15740 Z=-(A=0)                    <222>
15750 RETURN                      <058>
16090 :                           <062>
16100 REM *** AND ABSOLUTE,X      <176>
16110 GOSUB 4000                  <086>
16120 A=A AND PEEK(OO+Y)          <206>
16130 N=(A AND 128)/128           <149>
16140 Z=-(A=0)                    <114>
16150 RETURN                      <206>
16190 :                           <164>
16200 REM *** ROL ABSOLUTE,X      <226>
16210 GOSUB 4000                  <188>
16220 M=PEEK(OO+X)                <077>
16230 M=M*2+C                     <059>
16240 C=-(M>255)                  <051>
16250 M=M AND 255                 <230>
16260 POKE OO+X,M AND 255         <191>
16270 Z=-(M=0)                    <247>
16280 N=(M AND 128)/128           <172>
16290 RETURN                      <090>
16390 :                           <110>
16400 REM *** RTI                 <013>
16410 S=(S+1) AND 255             <246>
16420 P=ST%(S)                    <059>
16430 GOSUB 6000                  <170>
16440 S=(S+1) AND 255             <020>
16450 PC=ST%(S)                   <203>
16460 S=(S+1) AND 255             <040>
16470 PC=256*PC+ST%(S)            <121>
16480 RETURN                      <026>
16490 :                           <210>
16500 REM *** EOR (INDIRECT,X)    <045>
16510 GOSUB 3000                  <226>
16520 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                        <080>
16530 A=FN EO(PEEK(OO))           <021>
16540 N=(A AND 128)/128           <049>
16550 Z=-(A=0)                    <014>
16560 RETURN                      <106>
16890 :                           <100>
16900 REM *** EOR ZEROPAGE        <217>
16910 GOSUB 3000                  <118>
16920 A=FN EO(PEEK(O))            <073>
16930 N=(A AND 128)/128           <187>
16940 Z=-(A=0)                    <152>
16950 RETURN                      <244>
16990 :                           <202>
17000 REM *** LSR ZEROPAGE        <001>
17010 GOSUB 3000                  <218>
17020 M=PEEK(O)                   <098>
17030 C=M AND 1                   <193>
17040 M=INT(M/2)                  <235>
17050 POKE O,M                    <097>
17060 Z=-(M=0)                    <019>
17070 N=(M AND 128)/128           <200>
17080 RETURN                      <118>
17190 :                           <148>
17200 REM *** PHA                 <165>
17210 ST%(S)=A:S=(S-1) AND 255    <200>
17220 RETURN                      <004>
17290 :                           <248>
17300 REM *** EOR #IMMEDIATE      <222>
17310 GOSUB 3000                  <008>
17320 A=FN EO(O)                  <082>
17330 N=(A AND 128)/128           <077>
17340 Z=-(A=0)                    <042>
17350 RETURN                      <134>
17390 :                           <092>
17400 REM *** LSR A               <132>
17410 C=A AND 1                   <129>
17420 A=INT(A/2)                  <056>
17430 Z=-(A=0)                    <134>
17440 N=(A AND 128)/128           <076>
17450 RETURN                      <236>
17590 :                           <038>
17600 REM *** JMP ABSOLUTE        <172>
17610 GOSUB 4000                  <062>
17620 PC=OO-1                     <164>
17630 RETURN                      <160>
17690 :                           <140>
17700 REM *** EOR ABSOLUTE        <212>
17710 GOSUB 4000                  <164>
17720 A=FN EO(PEEK(OO))           <197>
17730 N=(A AND 128)/128           <225>
17740 Z=-(A=0)                    <190>
17750 RETURN                      <026>
17790 :                           <240>
17800 REM *** LSR ABSOLUTE        <252>
17810 GOSUB 4000                  <008>
17820 M=PEEK(OO)                  <196>
17830 C=M AND 1                   <231>
17840 M=INT(M/2)                  <017>
17850 POKE OO,M                   <232>
17860 Z=-(M=0)                    <057>
17870 N=(M AND 128)/128           <238>
17880 RETURN                      <156>
17990 :                           <186>
18000 REM *** BVC                 <080>
18010 GOSUB 3000                  <202>
18020 IF V=0 THEN GOSUB 7000      <136>
18030 RETURN                      <052>
18090 :                           <030>
18100 REM *** EOR (INDIRECT),Y    <015>
18110 GOSUB 3000                  <046>
18120 OO=PEEK(O)+256*PEEK(O+1)
      +Y                          <112>
18130 A=FN EO(PEEK(OO))           <097>
18140 N=(A AND 128)/128           <125>
18150 Z=-(A=0)                    <090>
18160 RETURN                      <182>
18490 :                           <178>
18500 REM *** EOR ZEROPAGE,X      <222>
18510 GOSUB 3000                  <194>
18520 A=FN EO(PEEK(O+X))          <003>
18530 N=(A AND 128)/128           <007>
18540 Z=-(A=0)                    <228>
18550 RETURN                      <064>
18590 :                           <022>
18600 REM *** LSR ZEROPAGE,X      <006>
18610 GOSUB 3000                  <038>
18620 M=PEEK(O+X)                 <025>
18630 C=M AND 1                   <013>
18640 M=INT(M/2)                  <055>
18650 POKE O+X,M                  <168>
18660 Z=-(M=0)                    <095>
18670 N=(M AND 128)/128           <020>
18680 RETURN                      <194>
18790 :                           <224>
18800 REM *** CLI                 <183>
18810 I=0                         <201>
18820 RETURN                      <080>
18890 :                           <068>
18900 REM *** EOR ABSOLUTE,Y      <073>
18910 GOSUB 4000                  <092>
18920 A=FN EO(PEEK(OO+Y))         <121>
18930 N=(A AND 128)/128           <153>
18940 Z=-(A=0)                    <118>
18950 RETURN                      <212>
19290 :                           <216>
19300 REM *** EOR ABSOLUTE,X      <217>
19310 GOSUB 4000                  <240>
19320 A=FN EO(PEEK(OO+X))         <237>
19330 N=(A AND 128)/128           <045>
19340 Z=-(A=0)                    <010>
19350 RETURN                      <102>
19390 :                           <060>
19400 REM *** LSR ABSOLUTE,X      <001>
19410 GOSUB 4000                  <084>
19420 M=PEEK(OO+X)                <229>
19430 C=M AND 1                   <051>
19440 M=INT(M/2)                  <093>
19450 POKE OO+X,M                 <042>
19460 Z=-(M=0)                    <135>
19470 N=(M AND 128)/128           <060>
19480 RETURN                      <234>
19590 :                           <006>
19600 REM *** RTS                 <175>
19610 S=(S+1) AND 255:HI=ST%(S
      )                           <147>
19620 S=(S+1) AND 255:LO=ST%(S
      )                           <161>
19630 PC=256*HI+LO                <221>
19640 REM  PC=PC+1 GESCHIEHT B
      EI 1040                     <116>
19650 RETURN                      <148>
19690 :                           <106>
19700 REM *** ADC (INDIRECT,X)    <048>
19710 GOSUB 3000                  <122>
19720 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                        <234>
19730 M=PEEK(OO)                  <074>
19740 GOSUB 8000                  <194>
19750 RETURN                      <250>
20090 :                           <254>
20100 REM *** ADC ZEROPAGE        <220>
20110 GOSUB 3000                  <014>
20120 M=PEEK(O)                   <150>
20130 GOSUB 8000                  <074>
20140 RETURN                      <130>
20190 :                           <098>
20200 REM *** ROR ZEROPAGE        <024>
20210 GOSUB 3000                  <114>
```

Listing »Demo 6502« (Fortsetzung)

```
20220 M=PEEK(O)                        <250>
20230 M=(128*C)+M/2                    <254>
20240 C=-(M<>INT(M))                   <024>
20250 M=M AND 255                      <166>
20260 POKE O,M                         <005>
20270 Z=-(M=0)                         <183>
20280 N=(M AND 128)/128                <108>
20290 RETURN                           <026>
20390 :                                <044>
20400 REM *** PHP                      <076>
20410 S=(S+1) AND 255                  <180>
20420 A=ST%(S)                         <189>
20430 RETURN                           <166>
20490 :                                <146>
20500 REM *** ADC #IMMEDIATE           <227>
20510 GOSUB 3000                       <162>
20520 M=O                              <134>
20530 GOSUB 8000                       <222>
20540 RETURN                           <022>
20590 :                                <246>
20600 REM *** ROR A                    <157>
20610 A=(128*C)+A/2                    <140>
20620 C=-(A<>INT(A))                   <051>
20630 A=A AND 255                      <052>
20640 Z=-(A=0)                         <040>
20650 N=(A AND 128)/128                <095>
20660 RETURN                           <142>
20790 :                                <192>
20800 REM *** JMP INDIRECT             <241>
20810 GOSUB 4000                       <216>
20820 PC=PEEK(OO)+256*PEEK(OO+
      1)-1                             <024>
20830 RETURN                           <058>
20890 :                                <036>
20900 REM *** ADC ABSOLUTE             <215>
20910 GOSUB 4000                       <060>
20920 M=PEEK(OO)                       <248>
20930 GOSUB 8000                       <112>
20940 RETURN                           <168>
20990 :                                <136>
21000 REM *** ROR ABSULUTE             <118>
21010 GOSUB 4000                       <162>
21020 M=PEEK(OO)                       <094>
21030 M=(128*C)+M/2                    <036>
21040 C=-(M<>INT(M))                   <062>
21050 M=M AND 255                      <204>
21060 POKE OO,M                        <140>
21070 Z=-(M=0)                         <221>
21080 N=(M AND 128)/128                <146>
21090 RETURN                           <064>
21190 :                                <082>
21200 REM *** BVS                      <248>
21210 GOSUB 3000                       <098>
21220 IF V=1 THEN GOSUB 7000           <064>
21230 RETURN                           <204>
21290 :                                <184>
21300 REM *** ADC (INDIRECT),Y         <020>
21310 GOSUB 3000                       <200>
21320 OO=PEEK(O)+256*PEEK(O+1)
      +Y                               <010>
21330 M=PEEK(OO)                       <150>
21340 GOSUB 8000                       <014>
21350 RETURN                           <070>
21690 :                                <074>
21700 REM *** ADC ZEROPAGE,X           <225>
21710 GOSUB 3000                       <090>
21720 M=PEEK(O+X)                      <077>
21730 GOSUB 8000                       <150>
21740 RETURN                           <206>
21790 :                                <176>
21800 REM *** ROR ZEROPAGE,X           <031>
21810 GOSUB 3000                       <192>
21820 M=PEEK(O+X)                      <179>
21830 M=(128*C)+M/2                    <074>
21840 C=-(M<>INT(M))                   <100>
21850 M=M AND 255                      <242>
21860 POKE O+X,M                       <076>
21870 Z=-(M=0)                         <003>
21880 N=(M AND 128)/128                <184>
21890 RETURN                           <102>
21990 :                                <120>
22000 REM *** SEI                      <207>
22010 I=1                              <113>
22020 RETURN                           <234>
22090 :                                <222>
22100 REM *** ADC ABSOLUTE,Y           <078>
22110 GOSUB 4000                       <246>
22120 M=PEEK(OO+Y)                     <137>
22130 GOSUB 8000                       <042>
22140 RETURN                           <098>
22490 :                                <112>
22500 REM *** ADC ABSOLUTE,X           <220>
22510 GOSUB 4000                       <136>
22520 M=PEEK(OO+X)                     <025>
22530 GOSUB 8000                       <190>
22540 RETURN                           <246>
22590 :                                <214>
22600 REM *** ROR ABSOLUTE,X           <026>
22610 GOSUB 4000                       <238>
22620 M=PEEK(OO+X)                     <127>
22630 M=(128*C)+M/2                    <112>
22640 C=-(M<>INT(M))                   <138>
22650 M=M AND 255                      <024>
22660 POKE OO+X,M AND 255              <241>
22670 Z=-(M=0)                         <041>
22680 N=(M AND 128)/128                <222>
22690 RETURN                           <140>
22890 :                                <004>
22900 REM *** STA (INDIRECT,X)         <084>
22910 GOSUB 3000                       <020>
22920 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                             <130>
22930 POKE OO,A                        <229>
22940 RETURN                           <136>
23190 :                                <050>
23200 REM *** STY ZEROPAGE             <178>
23210 GOSUB 3000                       <066>
23220 POKE O,Y                         <045>
23230 RETURN                           <172>
23290 :                                <150>
23300 REM *** STA ZEROPAGE             <000>
23310 GOSUB 3000                       <168>
23320 POKE O,A                         <144>
23330 RETURN                           <018>
23390 :                                <252>
23400 REM *** STX ZEROPAGE             <123>
23410 GOSUB 3000                       <012>
23420 POKE O,X                         <215>
23430 RETURN                           <118>
23590 :                                <198>
23600 REM *** DEY                      <106>
23610 Y=(Y-1) AND 255                  <103>
23620 Z=-(Y=0)                         <234>
23630 N=(Y AND 128)/128                <030>
23640 RETURN                           <074>
23790 :                                <142>
23800 REM *** TXA                      <168>
23810 A=X                              <219>
23820 Z=-(A=0)                         <174>
23830 N=(A AND 128)/128                <229>
23840 RETURN                           <122>
23990 :                                <088>
24000 REM *** STY ABSOLUTE             <173>
24010 GOSUB 4000                       <112>
24020 POKE OO,Y                        <053>
24030 RETURN                           <210>
24090 :                                <190>
24100 REM *** STA ABSOLUTE             <251>
24110 GOSUB 4000                       <214>
24120 POKE OO,A                        <149>
24130 RETURN                           <056>
24190 :                                <034>
24200 REM *** STX ABSOLUTE             <118>
24210 GOSUB 4000                       <058>
24220 POKE OO,X                        <191>
24230 RETURN                           <156>
24390 :                                <236>
24400 REM *** BCC                      <248>
24410 GOSUB 3000                       <252>
24420 IF C=0 THEN GOSUB 7000           <034>
24430 RETURN                           <102>
24490 :                                <080>
24500 REM *** STA (INDIRECT),Y         <054>
24510 GOSUB 3000                       <096>
24520 OO=PEEK(O)+256*PEEK(O+1)
      +Y                               <162>
24530 POKE OO,A                        <049>
24540 RETURN                           <212>
24790 :                                <126>
24800 REM *** STY ZEROPAGE,X           <183>
24810 GOSUB 3000                       <142>
24820 POKE O+X,Y                       <248>
24830 RETURN                           <248>
24890 :                                <228>
24900 REM *** STA ZEROPAGE,X           <085>
24910 GOSUB 3000                       <244>
24920 POKE O+X,A                       <082>
24930 RETURN                           <094>
24990 :                                <072>
25000 REM *** STX ZEROPAGE,Y           <132>
25010 GOSUB 3000                       <088>
25020 POKE O+Y,X                       <098>
25030 RETURN                           <194>
25190 :                                <018>
25200 REM *** TYA                      <172>
25210 A=Y                              <109>
25220 Z=-(A=0)                         <048>
25230 N=(A AND 128)/128                <103>
25240 RETURN                           <150>
25290 :                                <118>
25300 REM *** STA ABSOLUTE,Y           <112>
25310 GOSUB 4000                       <142>
25320 POKE OO+Y,A                      <122>
25330 RETURN                           <240>
25390 :                                <220>
25400 REM *** TXS                      <008>
25410 S=X                              <111>
25420 RETURN                           <076>
25690 :                                <010>
25700 REM *** STA ABSOLUTE,X           <000>
25710 GOSUB 4000                       <034>
25720 POKE OO+X,A                      <206>
25730 RETURN                           <132>
25990 :                                <056>
26000 REM *** LDY #IMMEDIATE           <098>
26010 GOSUB 3000                       <072>
26020 Y=O                              <092>
26030 N=(Y AND 128)/128                <144>
26040 Z=-(Y=0)                         <112>
26050 RETURN                           <198>
26090 :                                <156>
26100 REM *** LDA (INDIRECT,X)         <035>
26110 GOSUB 3000                       <172>
26120 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                             <028>
26130 A=PEEK(OO)                       <076>
26140 N=(A AND 128)/128                <253>
26150 Z=-(A=0)                         <218>
26160 RETURN                           <054>
26190 :                                <002>
26200 REM *** LDX #IMMEDIATE           <043>
26210 GOSUB 3000                       <018>
26220 X=O                              <034>
26230 N=(X AND 128)/128                <058>
26240 Z=-(X=0)                         <250>
26250 RETURN                           <144>
26390 :                                <204>
26400 REM *** LDY ZEROPAGE             <174>
26410 GOSUB 3000                       <220>
26420 Y=PEEK(O)                        <148>
26430 N=(Y AND 128)/128                <036>
26440 Z=-(Y=0)                         <004>
26450 RETURN                           <090>
26490 :                                <048>
26500 REM ** LDA ZEROPAGE              <004>
26510 GOSUB 3000                       <064>
26520 A=PEEK(O)                        <152>
26530 N=(A AND 128)/128                <133>
26540 Z=-(A=0)                         <098>
26550 RETURN                           <190>
26590 :                                <148>
26600 REM *** LDX ZEROPAGE             <074>
26610 GOSUB 3000                       <164>
26620 X=PEEK(O)                        <088>
26630 N=(X AND 128)/128                <206>
26640 Z=-(X=0)                         <142>
26650 RETURN                           <036>
26790 :                                <094>
26800 REM *** TAY                      <004>
26810 Y=A                              <152>
26820 Z=-(Y=0)                         <130>
26830 N=(Y AND 128)/128                <182>
26840 RETURN                           <226>
26890 :                                <196>
26900 REM *** LDA #IMMEDIATE           <214>
26910 GOSUB 3000                       <212>
26920 A=O                              <136>
26930 N=(A AND 128)/128                <025>
26940 Z=-(A=0)                         <246>
26950 RETURN                           <082>
26990 :                                <040>
27000 REM *** TAX                      <205>
27010 X=A                              <094>
27020 Z=-(X=0)                         <012>
27030 N=(X AND 128)/128                <096>
27040 RETURN                           <172>
27190 :                                <242>
27200 REM *** LDY ABSOLUTE             <126>
27210 GOSUB 4000                       <010>
27220 Y=PEEK(OO)                       <246>
27230 N=(Y AND 128)/128                <074>
27240 Z=-(Y=0)                         <042>
27250 RETURN                           <128>
27290 :                                <086>
27300 REM *** LDA ABSOLUTE             <202>
27310 GOSUB 4000                       <110>
27320 A=PEEK(OO)                       <250>
27330 N=(A AND 128)/128                <171>
27340 Z=-(A=0)                         <136>
27350 RETURN                           <228>
27390 :                                <186>
27400 REM *** LDX ABSOLUTE             <071>
27410 GOSUB 4000                       <212>
27420 X=PEEK(OO)                       <188>
27430 N=(X AND 128)/128                <244>
27440 Z=-(X=0)                         <180>
27450 RETURN                           <074>
27590 :                                <132>
27600 REM *** CLV                      <104>
27610 V=0                              <161>
27620 RETURN                           <244>
27690 :                                <234>
27700 REM *** LDA (INDIRECT),Y         <007>
27710 GOSUB 3000                       <250>
27720 OO=PEEK(O)+256*PEEK(O+1)
      +Y                               <060>
27730 A=PEEK(OO)                       <152>
27740 N=(A AND 128)/128                <073>
27750 Z=-(A=0)                         <038>
27760 RETURN                           <130>
27990 :                                <024>
28000 REM *** LDY ZEROPAGE,X           <136>
28010 GOSUB 3000                       <040>
28020 Y=PEEK(O+X)                      <075>
28030 N=(Y AND 128)/128                <112>
28040 Z=-(Y=0)                         <080>
28050 RETURN                           <166>
28090 :                                <124>
28100 REM *** LDA ZEROPAGE,X           <212>
28110 GOSUB 3000                       <140>
28120 A=PEEK(O+X)                      <079>
28130 N=(A AND 128)/128                <209>
28140 Z=-(A=0)                         <174>
28150 RETURN                           <010>
28190 :                                <226>
```

```
28200 REM *** LDX ZEROPAGE,X       <081>
28210 GOSUB 3000                   <242>
28220 X=PEEK(O+Y)                  <018>
28230 N=(X AND 128)/128            <026>
28240 Z=-(X=0)                     <218>
28250 RETURN                       <112>
28390 :                            <170>
28400 REM *** CLV                  <142>
28410 V=0                          <199>
28420 RETURN                       <028>
28490 :                            <016>
28500 REM *** LDA ABSOLUTE,Y       <065>
28510 GOSUB 4000                   <040>
28520 A=PEEK(OO+Y)                 <139>
28530 N=(A AND 128)/128            <101>
28540 Z=-(A=0)                     <066>
28550 RETURN                       <158>
28590 :                            <116>
28600 REM *** TSX                  <034>
28610 X=S                          <203>
28620 Z=-(X=0)                     <088>
28630 N=(X AND 128)/128            <172>
28640 RETURN                       <248>
28790 :                            <062>
28800 REM *** LDY ABSOLUTE,X       <131>
28810 GOSUB 4000                   <086>
28820 Y=PEEK(OO+X)                 <023>
28830 N=(Y AND 128)/128            <150>
28840 Z=-(Y=0)                     <118>
28850 RETURN                       <204>
28890 :                            <162>
28900 REM *** LDA ABSOLUTE,X       <207>
28910 GOSUB 4000                   <186>
28920 A=PEEK(OO+X)                 <027>
28930 N=(A AND 128)/128            <249>
28940 Z=-(A=0)                     <214>
28950 RETURN                       <050>
28990 :                            <008>
29000 REM *** LDX ABSOLUTE,Y       <080>
29010 GOSUB 4000                   <032>
29020 X=PEEK(OO+Y)                 <223>
29030 N=(X AND 128)/128            <064>
29040 Z=-(X=0)                     <000>
29050 RETURN                       <150>
29190 :                            <210>
29200 REM *** CPY #IMMEDIATE       <191>
29210 GOSUB 3000                   <226>
29220 C=-((Y-O)>=0)                <020>
29230 N=((Y-O) AND 128)/128        <057>
29240 Z=-((Y-O)=0)                 <057>
29250 RETURN                       <096>
29290 :                            <054>
29300 REM *** CMP (INDIRECT,X)     <013>
29310 GOSUB 3000                   <070>
29320 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                         <180>
29330 M=PEEK(OO)                   <020>
29340 GOSUB 8400                   <204>
29350 RETURN                       <196>
29590 :                            <100>
29600 REM *** CPY ZEROPAGE         <222>
29610 GOSUB 3000                   <116>
29620 M=PEEK(O)                    <252>
29630 C=-((Y-M)>=0)                <172>
29640 N=((Y-M) AND 128)/128        <211>
29650 Z=-((Y-M)=0)                 <209>
29690 :                            <200>
29700 REM *** CMP ZEROPAGE         <185>
29710 GOSUB 3000                   <218>
29720 M=PEEK(O)                    <098>
29730 GOSUB 8400                   <086>
29740 RETURN                       <078>
29790 :                            <046>
29800 REM *** DEC ZEROPAGE         <077>
29810 GOSUB 3000                   <062>
29820 M=PEEK(O)                    <198>
29830 M=(M-1) AND 255              <049>
29840 POKE O,M                     <187>
29850 Z=-(M=0)                     <109>
29860 N=(M AND 128)/128            <034>
29870 RETURN                       <208>
29990 :                            <248>
30000 REM *** INY                  <098>
30010 Y=(Y+1) AND 255              <089>
30020 Z=-(Y=0)                     <028>
30030 N=(Y AND 128)/128            <080>
30040 RETURN                       <124>
30090 :                            <092>
30100 REM *** CMP #IMMEDIATE       <190>
30110 GOSUB 3000                   <108>
30120 M=O                          <080>
30130 GOSUB 8400                   <232>
30140 RETURN                       <224>
30190 :                            <192>
30200 REM *** DEX                  <099>
30210 X=(X-1) AND 255              <063>
30220 Z=-(X=0)                     <166>
30230 N=(X AND 128)/128            <250>
30240 RETURN                       <147>
30390 :                            <138>
30400 REM *** CPY ABSOLUTE         <217>
30410 GOSUB 4000                   <162>
30420 M=PEEK(OO)                   <094>
30430 C=-((Y-M)>=0)                <210>
30440 N=((Y-M) AND 128)/128        <249>
30450 Z=-((Y-M)=0)                 <247>
30460 RETURN                       <034>
30490 :                            <240>
30500 REM *** CMP ABSOLUTE         <180>
30510 GOSUB 4000                   <008>
30520 M=PEEK(OO)                   <196>
30530 GOSUB 8400                   <124>
30540 RETURN                       <116>
30590 :                            <084>
30600 REM *** DEC ABSOLUTE         <072>
30610 GOSUB 4000                   <108>
30620 M=PEEK(OO)                   <040>
30630 M=(M-1) AND 255              <087>
30640 POKE OO,M                    <066>
30650 Z=-(M=0)                     <147>
30660 N=(M AND 128)/128            <072>
30670 RETURN                       <246>
30790 :                            <030>
30800 REM *** BNE                  <178>
30810 GOSUB 3000                   <046>
30820 IF Z=0 THEN GOSUB 7000       <012>
30830 RETURN                       <152>
30890 :                            <130>
30900 REM *** CMP (INDIRECT),Y     <239>
30910 GOSUB 3000                   <146>
30920 OO=PEEK(O)+256*PEEK(O+1)
      +Y                           <212>
30930 M=PEEK(OO)                   <096>
30940 GOSUB8400                    <024>
30950 RETURN                       <016>
31290 :                            <022>
31300 REM *** CMP ZEROPAGE,X       <190>
31310 GOSUB 3000                   <038>
31320 M=PEEK(O+X)                  <025>
31330 GOSUB 8400                   <162>
31340 RETURN                       <154>
31390 :                            <122>
31400 REM *** DEC ZEROPAGE,X       <082>
31410 GOSUB 3000                   <138>
31420 M=PEEK(O+X)                  <125>
31430 M=(M-1) AND 255              <125>
31440 POKE O+X,M                   <002>
31450 Z=-(M=0)                     <185>
31460 N=(M AND 128)/128            <110>
31470 RETURN                       <028>
31590 :                            <068>
31600 REM *** CLD                  <022>
31610 D=0                          <025>
31620 RETURN                       <180>
31690 :                            <168>
31700 REM *** CMP ABSOLUTE,Y       <041>
31710 GOSUB 4000                   <192>
31720 M=PEEK(OO+Y)                 <083>
31730 GOSUB 8400                   <052>
31740 RETURN                       <044>
32090 :                            <060>
32100 REM *** CMP ABSOLUTE,X       <185>
32110 GOSUB 4000                   <084>
32120 M=PEEK(OO+X)                 <229>
32130 GOSUB 8400                   <200>
32140 RETURN                       <192>
32190 :                            <160>
32200 REM *** DEC ABSOLUTE,X       <077>
32210 GOSUB 4000                   <184>
32220 M=PEEK(OO+X)                 <073>
32230 M=(M-1) AND 255              <163>
32240 POKE OO+X,M                  <132>
32250 Z=-(M=0)                     <223>
32260 N=(M AND 128)/128            <150>
32270 RETURN                       <068>
32390 :                            <106>
32400 REM *** CPX #IMMEDIATE       <086>
32410 GOSUB 3000                   <122>
32420 C=-((X-O)>=0)                <044>
32430 N=((X-O) AND 128)/128        <145>
32440 Z=-((X-O)=0)                 <081>
32450 RETURN                       <248>
32490 :                            <206>
32500 REM *** SBC (INDIRECT,X)     <023>
32510 GOSUB 3000                   <222>
32520 OO=PEEK(O+X)+256*PEEK(O+
      X+1)                         <078>
32530 M=PEEK(OO)                   <174>
32540 GOSUB 8200                   <070>
32550 RETURN                       <094>
32790 :                            <255>
32800 REM *** CPX ZEROPAGE         <041>
32810 GOSUB 3000                   <015>
32820 M=PEEK(O)                    <151>
32830 C=-((X-M)>=0)                <199>
32840 N=((C-M) AND 128)/128        <232>
32850 Z=-((X-M)=0)                 <236>
32860 RETURN                       <151>
32890 :                            <099>
32900 REM *** SBC ZEROPAGE         <196>
32910 GOSUB 3000                   <115>
32920 M=PEEK(O)                    <251>
32930 GOSUB 8200                   <207>
32940 RETURN                       <231>
32990 :                            <199>
33000 REM *** INC ZEROPAGE         <172>
33010 GOSUB 3000                   <215>
33020 M=PEEK(O)                    <095>
33030 M=(M+1) AND 255              <140>
33040 POKE O,M                     <086>
33050 Z=-(M=0)                     <008>
33060 N=(M AND 128)/128            <189>
33070 RETURN                       <107>
33190 :                            <145>
33200 REM *** INX                  <250>
33210 X=(X+1) AND 255              <206>
33220 Z=-(X=0)                     <117>
33230 N=(X AND 128)/128            <201>
33240 RETURN                       <021>
33290 :                            <247>
33300 REM *** SBC #IMMEDIATE       <203>
33310 GOSUB 3000                   <007>
33320 M=O                          <235>
33330 GOSUB 8200                   <099>
33340 RETURN                       <123>
33390 :                            <091>
33400 REM *** NOP                  <125>
33460 RETURN                       <243>
33590 :                            <037>
33600 REM *** CPX ABSOLUTE         <115>
33610 GOSUB 4000                   <061>
33620 M=PEEK(OO)                   <249>
33630 C=-((X-M)>=0)                <237>
33640 M=((X-M) AND 128)/128        <080>
33650 Z=-((X-M)>=0)                <093>
33660 RETURN                       <189>
33690 :                            <137>
33700 REM *** SBC ABSOLUTE         <191>
33710 GOSUB 4000                   <161>
33720 M=PEEK(OO)                   <093>
33730 GOSUB 8200                   <245>
33740 RETURN                       <013>
33790 :                            <237>
33800 REM *** INC ABSOLUTE         <169>
33810 GOSUB 4000                   <007>
33820 M=PEEK(OO)                   <195>
33830 M=(M+1) AND 255              <178>
33840 POKE OO,M                    <221>
33850 Z=-(M=0)                     <046>
33860 N=(M AND 128)/128            <227>
33870 RETURN                       <145>
33990 :                            <183>
34000 REM *** BEQ                  <210>
34010 GOSUB 3000                   <199>
34020 IF Z=1 THEN GOSUB 7000       <197>
34030 RETURN                       <049>
34090 :                            <029>
34100 REM *** SBC (INDIRECT),Y     <252>
34110 GOSUB 3000                   <045>
34120 OO=PEEK(O)+256*PEEK(O+1)
      +Y                           <111>
34130 M=PEEK(OO)                   <251>
34140 GOSUB 8200                   <147>
34150 RETURN                       <171>
34490 :                            <175>
34500 REM *** SBC ZEROPAGE,X       <201>
34510 GOSUB 3000                   <191>
34520 M=PEEK(O+X)                  <178>
34530 GOSUB 8200                   <027>
34540 RETURN                       <051>
34590 :                            <021>
34600 REM *** INC ZEROPAGE,X       <179>
34610 GOSUB 3000                   <037>
34620 M=PEEK(O+X)                  <024>
34630 M=(M+1) AND 255              <216>
34640 POKE O+X,M                   <157>
34650 Z=-(M=0)                     <084>
34660 N=(M AND 128)/128            <009>
34670 RETURN                       <183>
34790 :                            <221>
34800 REM *** SED                  <047>
34810 D=1                          <194>
34820 RETURN                       <079>
34890 :                            <067>
34900 REM *** SBC ABSOLUTE,Y       <054>
34910 GOSUB 4000                   <091>
34920 M=PEEK(OO+Y)                 <238>
34930 GOSUB 8200                   <175>
34940 RETURN                       <199>
35290 :                            <213>
35300 REM *** SBC ABSOLUTE,X       <196>
35310 GOSUB 4000                   <237>
35320 M=PEEK(OO+X)                 <126>
35330 GOSUB 8200                   <067>
35340 RETURN                       <091>
35390 :                            <059>
35400 REM *** INC ABSOLUTE,X       <174>
35410 GOSUB 4000                   <083>
35420 M=PEEK(OO+X)                 <228>
35430 M=(M+1) AND 255              <254>
35440 POKE OO+X,M                  <031>
35450 Z=-(M=0)                     <122>
35460 N=(M AND 128)/128            <047>
35470 RETURN                       <221>
35500 :                            <169>
49000 FOR HH=34768 TO 34783:RE
      AD H:POKE HH,H:NEXT:RETU
      RN                           <055>
50000 DATA 9,240,8,8,88,8,8,8,
      8,8,8,8,8,0,0,0              <123>

READY.
```

Listing »Demo 6502« (Schluß)

Der Kampf ums Überleben

Sie haben den Auftrag, mit Ihrem Kampfhubschrauber »HUEY«, ihren Vorposten Material zukommen zu lassen. Der Kampf ums Überleben beginnt gleich nach dem Start, denn bedenken Sie, »der Feind schläft nie«.

Sie sind Pilot eines Kampfhubschraubers und haben die Aufgabe, ein entferntes Ziel zu erreichen, um einen Ihrer Stützpunkte mit Vorräten zu versorgen. Ihr Weg geht mitten durch feindliches Gebiet; Flugzeuge, Hubschrauber und Boden-Luft-Raketen versuchen, Sie abzuschießen.

Zu Ihrer Verteidigung haben Sie Granatenwerfer und Wärmeraketen zur Verfügung.

Sind Sie trotz aller Behinderungen glücklich an das Ziel gelangt, so erhalten Sie je nach Abschußquote eine Auszeichnung.

Starten des Programms

Nach dem Eintippen und Speichern starten Sie das Programm mit »Run«.

Das Titelbild erscheint, und Sie werden nach ihrem Namen gefragt, der bis zu acht Zeichen lang sein darf.

Haben Sie diesen eingegeben, so werden Sie noch nach dem Schwierigkeitsgrad gefragt; hier geben Sie als »Neuling« am besten »1« ein. Jetzt erscheint Ihr Hubschraubercockpit. Der Hubschrauber steht auf der Erde, mit laufendem Motor.

Beschreibung der Instrumente

Oben links sehen Sie die fünf Leuchtanzeigen, die digital die Umdrehungen des Motors anzeigen. Die erste von links leuchtet; dies ist das Zeichen für »Motor eingeschaltet«. Rechts daneben ist die Statusanzeige für die vier Raketenrohre zu sehen (vier Leuchten, alle weiß).

Ganz rechts oben sind dann noch sechs System-Kontrolleuchten zu sehen (von links): System ein, Fuel, Granaten, Raketen, Rotor und Motor. Grün oder weiß bedeutet immer »OK«, rot bedeutet »Fehler«.

Weiter unten sehen Sie dann die Anzeige für den Granatenvorrat (»AMM.«) und für Raketen (»ROC.«).

Daneben die Geschwindigkeitsanzeige (»SPD.«), Höhenanzeige (»ALT.«), darunter die (analoge) Anzeige der Motorumdrehungen pro Minute (»RPM.«), der Rotorumdrehungen pro Minute (»ROT.«) und die Kilometer-Anzeige (»KM«), die Ihnen die zurückgelegte Strecke anzeigt. Rechts dann noch die Punktestand-Anzeige (»SCORE«) und in der Mittelkonsole der Kompaß (»POS.«) und die Treibstoffanzeige (»FUEL«).

Die Steuerung des Hubschraubers

Gesteuert wird der Hubschrauber über einen Joystick in Port 1 und die Tastatur.

»Joystickbewegungen«:

Joystick links	– Linkskurve des Hubschraubers
Joystick rechts	– Rechtskurve des Hubschraubers
Joystick oben	– Erhöhen der Fluggeschwindigkeit
Joystick unten	– Vermindern der Fluggeschwindigkeit
Joystick links + Feuer	– Erhöhen der Motordrehzahl
Joystick rechts + Feuer	– Vermindern der Motordrehzahl
Joystick oben + Feuer	– Vermindern der Flughöhe
Joystick unten + Feuer	– Erhöhen der Flughöhe
Feuer	– Abschuß einer Granate
»Tastatur:« 1,2,3,4	– Rakete in entsprechendes Rohr laden
[Space]	– Abschuß einer Rakete
[Return]	– Rotor einkuppeln
F	– Tank, Granaten und Raketen auffüllen (nur bei Höhe 0)

Der Start des Hubschraubers

Zunächst müssen Sie den Rotor einkuppeln. Drücken Sie hierzu so lange auf <RETURN>, bis die Systemkontrolle (oben rechts) für den Rotor grün zeigt. Der Rotor beginnt sich nun zu drehen (Schauen Sie auf die Anzeige für den Rotor [»ROT.«]).

Warten Sie, bis der Rotor 1/10 der Motorumdrehungen erreicht hat (die Anzeige für den Rotor steigt nicht mehr), dann steigern Sie die Umdrehungszahl des Motors bis auf mehr als 2000 rpm (Joystick links + Feuer) und warten wieder, bis die Rotordrehzahl gleich der Motordrehzahl ist. Bevor Sie jetzt abheben, sollten Sie die Raketenrohre laden (1 bis 4 drücken, die Anzeigen oben in der Mitte müssen rot leuchten), da, sobald Sie abheben, die Feinde kommen.

Die Feinde und ihre Bekämpfung

Hubschrauber werden am besten mit Granaten abgeschossen (Feuer drücken). Achtung: Nicht jede Granate trifft, daher muß man mehrmals auf einen Hubschrauber schießen; falls die fünfte Granate nicht trifft, sollte man lieber eine Rakete abschießen! Ein Hubschrauber bringt 50 Punkte. Flugzeuge können nur mit Raketen abgeschossen werden und bringen je 150 Punkte.

Raketen treten nur unter einer bestimmten, vom Level abhängigen Höhe auf (Level 1 = 2000 m, Level 2 = 4000 m etc.). Sie müssen ebenfalls mit Raketen abgeschossen werden und bringen 200 Punkte.

Fliegen des Hubschraubers

Zum Abheben drücken Sie Feuer und ziehen gleichzeitig den Joystick nach unten. Schauen Sie kurz auf die Höhenanzeige (»ALT.«), die ansteigt und gehen Sie auf etwa 1000 Fuß. Nun warten Sie, bis in Ihrem Sichtfeld ein Feind (als Punkt sichtbar) erscheint. Versuchen sie nun, den Feind durch Steigern der Höhe knapp über den Horizont zu bringen und durch Steuern links und rechts über den Pfeil (»↑«) auf dem Cockpit. Inzwischen dürfte der Feind so groß geworden sein, daß Sie erkennen, welche Art Sie vor sich haben. Wenn es kein Hubschrauber ist, feuern Sie eine Rakete ab, ansonsten drücken Sie mehrmals Feuer (Sie müssen den Hubschrauber so nah wie möglich herankommen lassen, sonst können Sie ihn nicht abschiessen).

Wenn dann gerade kein Feind in Sicht ist, können Sie losfliegen (Joystick nach oben drücken). Der Hubschrauber neigt sich nach vorne, die Geschwindigkeitsanzeige steigt, die Kilometer-Anzeige beginnt zu laufen.

Nach einer bestimmten Strecke (Level 1: 400 km, Level 2: 800 km) haben Sie es geschafft und werden bei entsprechendem Spielstand in die Hi-Score-Tabelle eingetragen.

Hinweis

Falls Sie keine Raketen/Granaten und/oder Mangel an Treibstoff haben (rote Leuchten in der Systemkontrolle), so kann der Hubschrauber nach dem Landen wieder aufgefüllt werden (F). Beachten Sie bitte, daß die Geschwindigkeit des Hubschraubers kleiner als 20 Meilen sein muß!

Die Positionsanzeige (»POS.«) gibt die Position des Hubschraubers in Grad an. Null Grad ist die Ideallinie; falls Sie unter einem Winkel fliegen, kommen Sie langsamer voran oder fliegen gar zurück (km-Anzeige.).

Bei manchen Joysticks (wie zum Beispiel Quickshot II) können Störungen auftreten!

Technische Daten des Hubschraubers

Geschwindigkeit: 260 Meilen/h (= 433 km/h)

Gipfelhöhe: 20000 Fuß (= 7,6 km)

Die geringe Gipfelhöhe erklärt sich durch die starke Bewaffnung!

(D. Kuhlmann/do)

```
10 REM *************
20 REM * HUEY C-16 *
30 REM *  1986 BY  *
40 REM *D. KUHLMANN*
60 REM *************
80 POKE55,255:POKE56,55:CLR
100 REM ***** VARIABLEN *****
110 N$(1)="PPPPPPPRRRRRRR*********EEEEEEEYYYYYYY"
120 N$(2)="FFFFFFFFFFFF************DDDDDDDDDDDD"
130 N$(3)="************************************"
140 N$(4)="DDDDDDDDDDDD************FFFFFFFFFFFF"
150 N$(5)="YYYYYYYEEEEEEE*********FFFFFFFPPPPPPP"
160 R$(1)="-----":R$(2)="CAPTAIN"
170 R$(3)="LIEUTNANT":R$(4)="COMMANDER"
180 R$(5)="GENERAL":R$(6)="{RVSON }RAMBO "
190 V=65280:AL=0:RO=0:RF=0:SP=0:RF=0
200 A=0:KO=0:P=0:KM=0:AA=200:RA=10
210 RP=800:FU=7000:N=3:X1=0:Y1=0:XA=0
220 YA=0:FL=0:G=0:AM=AA:RC=RA:RO=0
230 R(1)=0:R(2)=0:R(3)=0:R(4)=0:SC=0
240 E=0
250 REM ***** TITEL *****
260 S$="":POKEV+18,196:POKEV+19,208
270 COLOR0,1:COLOR4,1:COLOR1,2:SCNCLR
280 CHAR1,1,2,"KS-SOFTWARE SCHWEINFURT PRAESENTIER
T :"
290 CHAR1,2,5,"T T{2SPACE}T T{2SPACE}TTT{2SPACE}T
T{3SPACE}TTT{6SPACE}T TTT"
300 CHAR1,2,6,"T T{2SPACE}T T{2SPACE}T{4SPACE}T T{
3SPACE}T{8SPACE}T T"
310 CHAR1,2,7,"TTT{2SPACE}T T{2SPACE}TT{3SPACE}TTT
{3SPACE}T{3SPACE,SHFT-SPCE}TT{2SPACE}T TTT"
320 CHAR1,2,8,"T T{2SPACE}T T{2SPACE}T{5SPACE}T{4S
PACE}T{8SPACE}T T T"
330 CHAR1,2,9,"T T{2SPACE}TTT{2SPACE}TTT{3SPACE}T{
4SPACE}TTT{6SPACE}T TTT"
340 CHAR1,4,13,"WRITTEN 1986 BY D. KUHLMANN (C)"
350 CHAR1,0,18,"BITTE GEBEN SIE IHREN NAMEN EIN:"
360 GETKEYA$:A=ASC(A$):IFA=13GOTO440
370 L=LEN(S$):IFL>7GOTO440
380 IFA=20GOTO410
390 IFA<32ORA>90GOTO360
400 PRINTA$;:S$=S$+A$:GOTO360
410 IFL<1THEN360
420 S$=LEFT$(S$,L-1):PRINT"{LEFT,SPACE,LEFT}";:GOT
O360
430 :
440 IFLEN(S$)<8THENS$=S$+" ":GOTO440
450 CHAR1,0,20,"BITTE LEVEL (1 BIS 6) EINGEBEN !"
460 GETKEYA$:A=VAL(A$)
470 IFA<1ORA>6THEN460
480 L=A*2:KF=L*200:SCNCLR
490 REM ***** ZEICHENSATZ *****
500 RESTORE
510 POKEV+18,PEEK(V+18)AND251
520 POKEV+19,PEEK(V+19)OR56
530 FORI=832TO849:READA:POKEI,A:NEXT:SYS832
540 POKE836,210:POKE839,58
550 POKE842,211:POKE845,59:SYS832
560 DATA162,  0,189,  0,208,157,  0, 56
570 DATA189,  0,209,157,  0, 57,202,208
580 DATA241, 96
590 REM ***** NEUE ZEICHEN *****
600 FORI=14600TO14703:READQ:POKEI,Q:NEXT
610 DATA  0,  0,126,126,126,  0,  0,  0
620 DATA  0,  0,  0, 16, 16,  0,  0,  0
630 DATA  0,  0, 16, 56, 16,  0,  0,  0
640 DATA  0,  0, 16,124, 16,  0,  0,  0
650 DATA  0,  0, 24,255, 24,  0,  0,  0
660 REM RAKETE
670 DATA  0,  8,  8, 28, 28, 62,127,119
680 DATA  0,  0,  8, 28, 28, 62, 62, 62
690 DATA  0,  0,  0,  0,  0,  8, 28, 28
700 REM EXPLOSION
710 DATA164,137, 50, 73,168, 84, 44,129
720 REM HUBSCHRAUBER
730 DATA  0,  0,124, 56, 56,124, 68,  0
740 DATA255, 24,126,219,255,255,189,195
750 REM FEINDRAKETE
760 DATA119,127, 62, 28, 28,  8,  8,  0
770 DATA  0,  0, 28, 62, 62, 28,  0,  0
780 REM ***** VOREINSTELLUNGEN *****
790 VOL8:PUDEF"0"
800 PRINTCHR$(142)CHR$(8)
810 REM ***** BILD *****
820 POKEV+6,0
830 COLOR0,2,6:COLOR4,2,4
840 PRINT"{CLR,BLACK}"TAB(5)"¥{RVSON,5SPACE}0 H F{
2SPACE}1234{2SPACE}SFG&RM{4SPACE,RVOFF}£"
850 PRINTTAB(6)"¥{RVSON,2SPACE}H{RVOFF,WHITE}E{GRE
EN}!!!{ORANGE}!{RED}!{BLACK}={WHITE}R!!!!{BLACK}={
WHITE}C!!!!!!{BLACK,RVSON}G{2SPACE,RVOFF}£"
860 PRINTTAB(7)"¥{RVSON,2SPACE}M@@@@@@@@@@@@@@@@@@
N{2SPACE,RVOFF}£"
870 FORI=1TO8
880 PRINTTAB(8-I)"{RVSON}£{RVOFF}£"SPC(20+I*2)"¥{R
VSON}¥{RVOFF}"
890 NEXT
900 PRINT"{RVSON,5SPACE}¥{RVOFF}"SPC(28)"{RVSON}£{
5SPACE}"
910 PRINT"{RVSON,10SPACE}¥{RVOFF}"SPC(18)"{RVSON}£
{10SPACE}"
920 PRINT"{RVSON }AMM. 000{2SPACE,2SHFT-SPCE,6SPAC
E}↑{14SPACE}00000 "
930 PRINT"{RVSON }ROC.{2SPACE}00 SPD{3SPACE}000{2S
PACE}ALT 00000{4SPACE}SCORE "
940 PRINT"{RVSON,10SPACE}RPM{2SPACE}0000{2SPACE}RO
T{3SPACE}000{10SPACE}"
950 PRINT"{RVSON,10SPACE}KM{3SPACE}0000{21SPACE}"
960 PRINTTAB(6)"¥{RVSON,26SPACE,RVOFF}£"
970 PRINTTAB(10)"¥{RVSON,5SPACE}POS.{3SPACE}0{5SPA
CE,RVOFF}£"
980 PRINTTAB(13)"¥{RVSON,12SPACE,RVOFF}£"
990 PRINTTAB(14)"{RVSON,4SPACE}FUEL{4SPACE,RVOFF}"
1000 PRINTTAB(14)"{RVSON,3SPACE,RVOFF,RED}!{GREEN}
!!!!!{BLACK,RVSON,3SPACE,RVOFF}"
1010 PRINTTAB(14)"{RVSON,12SPACE,RVOFF}"
1020 PRINTTAB(14)"{RVSON,12SPACE,RVOFF}"
1030 PRINTTAB(13)"{RVSON}£{12SPACE}¥{RVOFF,HOME}"
1040 POKEV+6,16
1050 REM AKTUELLE WERTE PRINTEN
1060 CHAR1,6,13,"{BLACK,RVSON}":PRINTUSING"###";AM
1070 CHAR1,7,14,"{BLACK,RVSON}":PRINTUSING"##";RC
1080 CHAR1,15,15,"{BLACK,RVSON}":PRINTUSING"####";
RP
1090 CHAR1,34,13,"{BLACK,RVSON}":PRINTUSING"#####"
;SC
1100 CHAR1,27,15,"{BLACK,RVSON}":PRINTUSING"###";R
O
1110 COLOR1,6,5:CHAR1,11,1,"!":CHAR1,29,1,"!"
1120 CHAR1,24,1,"!"
1130 IFRF=1THENCHAR1,28,1,"!"
1140 REM ***** HAUPTPROGRAMM *****
1150 :
1160 SOUND1,0,0:SOUND2,0,0:Z=RP/30
1170 SOUND1,Z,150:SOUND2,Z+1,150
1180 REM HORIZONT
1190 COLOR1,10,2:CHAR1,2,10,N$(N)
1200 A=INT(RP/10):IFRO=AGOTO1250
1210 IFRF=0GOTO1250
1220 IFRO>ATHENRO=RO-1:GOTO1240
1230 RO=RO+1+INT(A/100)
1240 CHAR1,27,15,"{BLACK,RVSON}":PRINTUSING"###";R
O
1250 REM JOYSTICKABFRAGE
1260 A=JOY(1)
1270 IFA=7GOTO2010
1280 IFA=3GOTO2090
1290 IFA=1GOTO2160
1300 IFA=5GOTO2230
1310 IFA=133GOTO2290
1320 IFA=129GOTO2370
1330 IFA=135GOTO2520
1340 IFA=131GOTO2610
1350 IFA=128GOTO3050
1360 REM TASTATUR
1370 GETA$:IFA$=""GOTO1420
1380 IFA$=" "GOTO2740
1390 IFA$=CHR$(13)GOTO2710
1400 IFA$="F"ANDAL<1GOTO3190
1410 A=VAL(A$):IFA>0ANDA<5GOTO2440
1420 IFN<3GOTO2010:REM NEIGUNG LINKS
1430 IFN>3GOTO2090:REM NEIGUNG RECHTS
1440 REM FEIND
1450 IFE<>0GOTO1560:REM FEIND VORHANDEN
1460 IFAL<1GOTO1820:REM BODEN
1470 MV=0:E=INT(L*10*RND(0))
1480 IFE<18THENE=0:GOTO1820:REM KEIN F.
1490 FL=FL+1:E=INT(20*RND(1))
1500 X1=INT(26*RND(1))+7:XA=X1
1510 Y1=INT( 5*RND(1))+4:YA=Y1
1520 IFE<L*2ANDAL<L*1000THENE=2:GOTO1820:REM RAKET
E
1530 IFE<L*5THENE=3:GOTO1820:REM FLUGZ.
1540 E=1:GOTO1820:REM HUBSCHR.
1550 :
1560 REM FEIND BEWEGEN
```

Listing. »HUEY« (Fortsetzung)

```
1570 MV=MV+1:IFMV>64-L*4GOTO3460
1580 ONEGOTO1700,1740:REM FLUGZ.,RAKETE
1590 IFMV<16-LTHENZ=34:GOTO1630
1600 IFMV<32-L*2THENZ=35:GOTO1630
1610 IFMV<48-L*3THENZ=36:GOTO1630
1620 Z=37
1630 IFY1<3GOTO1980:REM AUS
1640 IFY1>9THENY1=9
1650 IFX1<12-Y1GOTO1980:REM AUS
1660 IFX1>27+Y1GOTO1980
1670 COLOR1,1:CHAR1,XA,YA," "
1680 CHAR1,X1,Y1,CHR$(Z)
1690 XA=X1:YA=Y1:GOTO1820
1700 IFMV<16-LTHENZ=34:GOTO1630
1710 IFMV<32-L*2THENZ=35:GOTO1630
1720 IFMV<48-L*3THENZ=42:GOTO1630
1730 Z=43:GOTO1630
1740 REM RAKETE
1750 COLOR1,1
1760 IFMV>52-L*4GOTO3240
1770 IFMV<13-LTHENCHAR1,19,9,CHR$(34):GOTO1820
1780 IFMV<26-L*2THENCHAR1,19,9,"#":GOTO1820
1790 IFMV<39-L*3THENCHAR1,19,9,"-":GOTO1820
1800 CHAR1,19,9," ":CHAR1,19,11,","
1810 :
1820 REM FUEL VERM.
1830 FU=FU-RP/500-AL/1000
1840 IFFU<1000THEN3240
1850 COLOR1,6,5:IFFU<2000GOTO1880
1860 CHAR1,16+FU/1000,21,"!"
1870 GOTO1890
1880 COLOR1,3,5:CHAR1,25,1,"!":GOTO1860
1890 REM KM PLUS
1900 IFKM>KFGOTO3540:REM AM ZIEL
1910 IFKO<90THENP=(90-KO)/2*SP:GOTO1950
1920 IFKO<180THENP=-(KO-90)/2*SP:GOTO1950
1930 IFKO<270THENP=-(270-KO)/2*SP:GOTO1950
1940 P=(KO-270)/2*SP
1950 KM=KM+P/10000:IFKM<0THEN3240
1960 PRINT:CHAR1,15,16,"{BLACK,RVSON}":PRINTUSING"
####";KM
1970 GOTO1140
1980 REM FEIND AUS
1990 PRINT:E=0:CHAR1,XA,YA," "
2000 GOTO1140
2010 REM ***** UNTERPROGRAMME *****
2020 REM LINKS FLIEGEN
2030 IFRF=0THEN1440:REM ROTOR NOCH AUS
2040 IFN<5THENN=N+1
2050 KO=KO-1:X1=X1+1:IFKO<0THENKO=359
2060 CHAR1,21,18,"{BLACK,RVSON}":PRINTUSING"###";K
O
2070 GOTO1440
2080 :
2090 REM RECHTS FLIEGEN
2100 IFRF=0THEN1440
2110 IFN>1THENN=N-1
2120 KO=KO+1:X1=X1-1:IFKO>359THENKO=0
2130 CHAR1,21,18,"{BLACK,RVSON}":PRINTUSING"###";K
O
2140 GOTO1440
2150 :
2160 REM SCHNELLER
2170 IFAL<1GOTO1360
2180 IFSP>250GOTO1360
2190 A=PEEK(V+6):IFA<23THENPOKEV+6,A+1
2200 SP=SP+10:CHAR1,16,14,"{BLACK,RVSON}":PRINTUSI
NG"###";SP
2210 GOTO1360
2220 :
2230 REM LANGSAMER
2240 IFSP<1GOTO1360
2250 A=PEEK(V+6):IFA>16THENPOKEV+6,A-1
2260 SP=SP-10:CHAR1,16,14,"{BLACK,RVSON}":PRINTUSI
NG"###";SP
2270 GOTO1360
2280 :
2290 REM STEIGEN
2300 IFRF=0ORRO<200THEN1360
2310 S=INT((20000-AL+RP)/2000)
2320 IFRP<AL/3THENS=S-(AL/150)
2330 IFS<.05THEN1360
2340 AL=AL+S:CHAR1,25,14,"{BLACK,RVSON}":PRINTUSIN
G"#####";AL
2350 Y1=Y1+1:GOTO1360
2360 :
2370 REM SINKEN
2380 AL=AL-20:IFAL<20THENAL=0
2390 CHAR1,25,14,"{BLACK,RVSON}":PRINTUSING"#####"
;AL
2400 IFAL<>0GOTO2420
2410 IFSP>10GOTO3240
2420 Y1=Y1-1:GOTO1360
2430 :
2440 REM RAKETE LADEN
2450 IFR(A)=1ORRC<1GOTO1360
2460 SOUND1,0,0:SOUND1,900,5
2470 R(A)=1:CHAR1,17+A,1,"{RVOFF,RED}!"
2480 RC=RC-1:CHAR1,7,14,"{BLACK,RVSON}":PRINTUSING
"##";RC
2490 IFRC<1THENCOLOR1,3,5:CHAR1,27,1,"!"
2500 GOTO1360
2510 :
2520 REM MOTOR SCHNELLER
2530 IFRP>9000THEN1360
2540 RP=RP+RP/80
2550 CHAR1,15,15,"{BLACK,RVSON}":PRINTUSING"####";
RP
2560 A=INT(RP/2200):COLOR1,6,5
2570 IFA=3THENCOLOR1,9,5
2580 IFA=4THENCOLOR1,3,5
2590 CHAR1,11+A,1,"!":GOTO1360
2600 :
2610 REM MOTOR LANGSAMER
2620 IFRP<801THEN1360
2630 RP=RP-RP/80
2640 CHAR1,15,15,"{BLACK,RVSON}":PRINTUSING"####";
RP
2650 A=INT(RP/2200):COLOR1,6,3
2660 IFA>3GOTO1140
2670 IFA=3THENCOLOR1,3,3
2680 IFA=2THENCOLOR1,9,3
2690 CHAR1,12+A,1,"!":GOTO1360
2700 :
2710 REM ROTOR ON
2720 COLOR1,6,5:CHAR1,28,1,"!"
2730 RF=1:GOTO1360
2740 REM RAKETE
2750 I=0:COLOR1,3,5
2760 I=I+1:IFR(I)=1THEN2780
2770 IFI<4THEN2760:ELSE1360
2780 SOUND1,0,0:SOUND2,0,0:SOUND2,900,10
2790 FORI=500TO0STEP-10
2800 SOUND3,I,1
2810 IFI=500THENCHAR1,19,12,"{RED,RVOFF}&"
2820 IFI=350THENCHAR1,19,12," ":CHAR1,19,11,"'"
2830 IFI=200THENCHAR1,19,11," ":CHAR1,19,10,"("
2840 NEXT
2850 CHAR1,19,10,"{ORANGE}*"
2860 SOUND3,0,10:I=0
2870 I=I+1:IFR(I)=1GOTO2880:ELSE2870
2880 R(I)=0:CHAR1,17+I,1,"{RVOFF,WHITE}!"
2890 IFPEEK(3451)<>32THEN2920
2900 GOTO1360
2910 :
2920 REM TREFFER
2930 G=G+1:B=PEEK(3451)
2940 SOUND3,0,0
2950 CHAR1,19,9,"{ORANGE})"
2960 SOUND3,10,100
2970 FORI=8TO0STEP-1:FORA=1TO100:NEXT
2980 VOLI:NEXT
2990 CHAR1,19,9," "
3000 SC=SC+50*E:IFE=2THENSC=SC+100
3010 E=0:VOL8
3020 CHAR1,34,13,"{BLACK,RVSON}":PRINTUSING"#####"
;SC
3030 GOTO1140
3040 :
3050 REM KANONE
3060 IFAM<1GOTO1360
3070 SOUND1,0,0:SOUND3,0,0:SOUND3,500,30
3080 FORI=8TO1STEP-1:FORA=1TO50:NEXT
3090 VOLI:NEXT:VOL8
3100 AM=AM-1:PRINT:CHAR1,6,13,"{BLACK,RVSON}"
3110 PRINTUSING"###";AM
3120 IFAM<1THENCOLOR1,3,5:CHAR1,26,1,"!"
3130 IFPEEK(3451)=43GOTO3150
3140 GOTO1440
3150 A=INT(10*RND(1))
3160 IFA<8GOTO1440
3170 GOTO2920
3180 :
3190 REM AUFFUELLEN
3200 SOUND1,0,0:SOUND1,600,20
3210 FU=7000:RC=RA:AM=AA
3220 FORI=1TO4:R(I)=0:NEXT
3230 GOTO810
```

Listing. »HUEY« (Fortsetzung)



```
3240 REM ***** GAME OVER *****
3250 SOUND3,0,0:SOUND3,100,100
3260 FORI=1TO16:FORA=0TO7:COLOR0,I,A
3270 NEXT:VOL(17-I)/2:NEXT
3280 PRINT"{BLACK}":CHAR1,15,7,"GAME OVER."
3290 CHAR1,14,9,"-PRESS FIRE-"
3300 IFJOY(1)<>128THEN3300
3310 REM HISCORE-TABLE
3320 SCNCLR:COLOR0,2,4:COLOR4,1
3330 CHAR1,13,2,"HISCORE-TABLE"
3340 CHAR1,8,4,"DIE 10 BESTEN PILOTEN:"
3350 FORI=1TO10:IFSC<=SC(I)THENNEXT:GOTO3400
3360 Z=I
3370 FORA=ITO9:SC(I+1)=SC(I)
3380 S$(I+1)=S$(I):NEXT
3390 S$(Z)=S$:SC(Z)=SC
3400 :
3410 FORI=1TO10:CHAR1,10,5+I,""
3420 PRINTUSING"##";I;:PRINT" "S$(I)SC(I)
3430 NEXT
3440 GETKEYA$:IFA$<>" "THEN3440
3450 GOTO100
3460 REM ***** ABSCHUSS *****
3470 FORI=1TO60
3480 SOUND3,0,0:SOUND3,1000,1
3490 FORA=1TO5:NEXT
3500 A1=INT(39*RND(1)):A2=INT(24*RND(1))
3510 PRINT:CHAR1,A1,A2,"{BLACK}@"
3520 IFI/10=INT(I/10)THENCHAR1,23+I/10,1,"{RED}!"
3530 NEXT:GOTO3240
3540 REM ***** GEWONNEN *****
3550 PRINT:SOUND1,0,0:SOUND2,0,0
3560 POKEV+18,196:POKEV+19,208
3570 IFE=1ANDFL>2THENFL=FL-1
3580 PRINT"{CLR}":A=100/FL*G/10-4:IFA<1THENA=1
3590 CHAR1,1,5,"{BLACK}SIE HABEN ES GESCHAFFT !"
3600 CHAR1,1,8,"SIE HABEN"+STR$(G)+" FEINDE ABGESC
HOSSEN !"
3610 CHAR1,1,11,"IHR RANG:"+R$(A)
3620 CHAR1,15,20,"[SPACE] !"
3630 FORA=1TO3:SOUND3,950,5:FORZ=1TO400:NEXT:NEXT
3640 FORA=1TO3:SOUND3,950,3:FORZ=1TO150:NEXT:NEXT
3650 GETA$:IFA$<>" "THEN3630
3660 GOTO3320
```

©64'er

Listing. »HUEY« (Schluß).
Bitte beachten Sie die Eingabehinweise auf Seite 76.

Raumschlacht auf dem VC 20

Kämpfen Sie sich durch einen Asteroidengürtel. Nur so können Sie mit Ihren fünf Raumschiffen überleben.

Asteroids ist ein typisches Action-Spiel in Maschinensprache. Der Spieler steuert ein Raumschiff in der Mitte des Spielfeldes, das sich um seine eigene Achse in alle Richtungen drehen läßt. Ihre Aufgabe ist es, zu verhindern, daß das Raumschiff durch herannahende Asteroiden zerstört wird. Es gilt nun, diese vorher durch Abschießen zu zerstören. Ab und zu verstecken sich auch Ufos unter den Asteroiden, deren Abschuß eine besonders hohe Punktzahl bringt. Am Anfang erscheinen nur einige große Asteroiden. Je besser der Spieler nun ist, desto mehr Asteroiden erscheinen – nun auch kleinere – und desto schneller wird das Spiel. Man hat fünf Raumschiffe zur Verfügung. Bei über 1000 Punkten gibt es Bonus-Raumschiffe. Ist ein Abschießen eines herannahenden Asteroiden nicht mehr möglich, kann mit der SPACE-Taste (Hyperspace) ein Raumsprung in die obere Bildschirmebene vollzogen werden (beziehungsweise wieder in die Position in der Bildschirmmitte). Dabei besteht jedoch das Risiko, daß Ihr Raumschiff zerstört wird. – Mit der Taste »F7« wird der Antrieb des Raumschiffes eingeschaltet. Er bewirkt, daß sich das Raumschiff langsam in die jeweilige Schußrichtung bewegt.

Eingabe des Programms:

Das Programm benötigt mindestens eine 8 KByte Erweiterung. Mit ihr wird das Programm wie folgt geladen:
Im Direktmodus bitte folgendes eingeben:

```
POKE 44,32:POKE 8192,0:NEW:POKE 648,30:SYS 58648.
```

Dann wird das Programm »Asteroids Loader« (Listing 1) eingetippt und gespeichert beziehungsweise geladen. Es enthält Prüfsummen, die auf einen Eingabefehler aufmerksam machen.

Jetzt wird mit dem Befehl RUN das Programm »Loader« gestartet. Wenn es dazu auffordert, ist NEW und <RETURN> einzugeben. Anschließend ist noch die Zeile

```
POKE 44,33:POKE 8448,0:NEW
```

erforderlich. Nun läßt sich das Hauptprogramm »Main Program« (Listing 2) eingeben und speichern, beziehungsweise von Kassette laden. Dieses startet man einfach mit RUN. Alles weitere wird im Programm erklärt.

Mit einer 16 KByte-Erweiterung ist vor dem Laden des Loader-Programms die Zeile:

```
POKE 44,33: POKE 8448,0:NEW:POKE 648,30: SYS 58648
```

einzugeben. Anschließend tippt man das Programm »Loader« ab und startet es mit RUN. Wenn das Programm dazu auffordert, gibt man NEW ein und lädt das Hauptprogramm ohne zusätzliche Befehle.

Durch Änderung des Wertes 70 in Zeile 12040 kann die Geschwindigkeit verändert werden (bis ...85). Wenn das Spiel nicht so schnell werden soll, kann dies geändert werden, indem im Hauptprogramm in Zeile 20010 der zweite Wert (3) erhöht wird, zum Beispiel auf 15.

(Claus Neupert/ah)

```
1 REM ***************************************
  **                                      <052>
2 REM                                     <064>
3 REM >> ASTEROIDS << [DATA-LOADER]       <200>
4 REM                                     <066>
5 REM                                     <067>
6 REM 1985 BY C. NEUPERT                  <140>
7 REM                                     <069>
8 REM                                     <070>
9 REM ***************************************
  **                                      <060>
10 DIM H(75):FOR I=0 TO 9                 <139>
20 H(48+I)=I:H(65+I)=I+10:NEXT I          <104>
30 FOR I=5420 TO 7399: READ A$            <231>
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1)) <223>
50 D=H(H)*16+H(L): S=S+D: POKE I,D        <243>
60 A=A+1: IF A<20 THEN NEXT: A=-1         <165>
65 PRINT "ZEILE ";1000+Z;                 <181>
70 READ V: Z=Z+1:IF V=S THEN 85           <049>
80 PRINT" FEHLER !";999+Z:STOP            <023>
85 IF A<0 THEN 100                        <223>
90 S=0:A=0:PRINT:NEXT                     <050>
100 PRINT"{CLR,2DOWN}GEBEN SIE JETZT 'NEW'
   ":PRINT"{DOWN}EIN UND LADEN SIE DAS":  <136>
110 PRINT"{DOWN}HAUPTPROGRAMM !"          <080>
```

Listing 1. Ladeprogramm für das Spiel »Asteroids«

```
120 END                                              <122>
990 REM                                              <034>
991 REM DECIMAL ADDRESS 5420 TO 7399                 <035>
992 REM                                              <036>
1000 DATA AD,58,1D,D0,01,60,CE,42,1D,30,01
     ,60,A9,19,8D,42,1D,20,52,17, 1608               <120>
1001 DATA A9,20,91,FB,AE,E4,03,BD,E6,03,20
     ,DA,1B,A5,FC,30,1D,C9,1E,30, 2474               <087>
1002 DATA 19,C9,20,10,15,B1,FB,8D,41,1D,AD
     ,E4,03,91,FB,A5,FB,8D,DC,03, 2538               <179>
1003 DATA A5,FC,8D,DD,03,60,20,52,17,18,90
     ,EA,DA,1B,A5,FC,C9,1E,30,18, 2382               <108>
1004 DATA0A5,C5,C9,3F,D0,03,8D,58,1D,C9,20
     ,D0,04,8D,56,1D,60,AD,E2,03, 2294               <250>
1005 DATA 85,00,20,2C,1A,20,00,19,AD,56,1D
     ,C9,02,D0,01,60,20,26,16,20, 1212               <176>
1006 DATA 52,17,B1,FB,C9,08,30,07,A9,03,8D
     ,56,1D,60,EA,A4,FE,A2,96,CA, 2487               <011>
1007 DATA D0,FD,88,D0,F8,20,2C,15,AD,41,1D
     ,C9,20,D0,E5,A4,00,C0,01,F0, 2684               <164>
1008 DATA 07,88,84,00,18,90,BF,EA,20,58,1B
     ,20,44,16,20,26,16,18,90,9C, 1553               <031>
1009 DATA AE,DF,03,D0,01,60,BD,3C,03,85,FB
     ,BD,64,03,85,FC,A9,20,A0,00, 2379               <103>
1010 DATA 91,FB,BD,8C,03,20,37,1B,F0,0B,8E
     ,F0,03,A0,00,8C,DF,03,4C,E5, 2309               <195>
1011 DATA 18,4C,D8,18,A0,00,A9,15,91,FB,A5
     ,FB,9D,3C,03,A5,FC,9D,64,03, 2399               <108>
1012 DATA CE,0B,90,60,90,60,8C,0B,90,31,AE
     ,E3,03,F0,04,CE,E3,03,60,A2, 2383               <091>
1013 DATA 03,8E,E3,03,AD,0D,90,30,01,60,38
     ,E9,04,8D,0D,90,60,EA,EA,EA, 2239               <012>
1014 DATA A9,F2,CD,0A,90,D0,06,A9,BF,8D,0A
     ,90,60,8D,0A,90,60,8D,0A,90, 2421               <005>
1015 DATA 60,31,36,30,20,A5,FB,48,A5,FC,48
     ,A2,20,A9,00,20,E8,1A,A2,1A, 2097               <069>
1016 DATA A9,16,20,E8,1A,A2,1A,A9,D4,20,E8
     ,1A,A2,17,A9,17,20,E8,1A,A2, 2169               <172>
1017 DATA 17,A9,FE,20,E8,1A,A2,1A,A9,18,20
     ,E8,1A,A2,17,A9,D2,20,E8,1A, 2261               <135>
1018 DATA A2,1A,A9,2C,20,E8,1A,A2,17,A9,D6
     ,20,E8,1A,68,85,FC,68,85,FB, 2526               <129>
1019 DATA 60,EA,EA,EA,EA,38,E9,0C,AA,BD,4C
     ,1D,20,DA,1B,20,C7,1A,A5,FC, 2748               <177>
1020 DATA 30,39,C9,1E,30,35,A0,04,A2,00,86
     ,01,20,5D,16,A9,9B,8D,0D,90, 1667               <254>
1021 DATA A9,0F,8D,38,1D,A2,DC,CA,D0,FD,C8
     ,20,5D,16,CE,38,1D,D0,F2,A9, 2712               <083>
1022 DATA 20,85,01,A0,01,20,5D,16,A9,19,8D
     ,0C,90,8D,0B,90,20,A8,02,60, 1559               <163>
1023 DATA C9,15,D0,06,A9,03,8D,56,1D,60,C9
     ,0C,30,03,4C,AD,16,20,C7,1A, 1752               <111>
1024 DATA A5,FC,30,33,C9,1E,30,2F,A0,00,A9
     ,17,91,FB,A9,02,20,F8,1A,A9, 2236               <113>
1025 DATA B7,8D,0D,90,A9,28,8D,38,1D,A2,FF
     ,CA,D0,FD,CE,38,1D,D0,F6,A9, 2910               <051>
1026 DATA 46,20,A8,02,A9,01,20,F8,1A,A9,20
     ,A0,00,91,FB,60,AD,F0,03,30, 2065               <251>
1027 DATA 08,CD,E2,03,10,03,4C,C0,17,60,AD
     ,DC,03,85,FB,AD,DD,03,85,FC, 2410               <090>
1028 DATA A0,00,60,36,33,A0,00,8C,0C,90,8C
     ,0B,90,4C,F8,16,EE,48,D1,8C, 2117               <027>
1029 DATA E8,E0,14,30,02,A2,00,60,EA,EA,E8
     ,E0,03,30,02,A2,00,60,EA,EA, 2487               <202>
1030 DATA EE,4B,1D,AE,4B,1D,E0,05,30,03,18
     ,90,1B,EE,FB,03,AE,FB,03,E0, 2239               <236>
1031 DATA 05,30,0B,A2,00,8E,FB,03,A9,0C,99
     ,B4,03,60,A9,08,99,B4,03,60, 1844               <150>
1032 DATA 4C,C4,18,A2,00,8E,4B,1D,A9,15,99
     ,B4,03,AD,F0,03,8D,DF,03,60, 2109               <108>
1033 DATA 4C,8A,18,E0,04,30,02,A2,00,E8,8E
     ,FA,03,E0,01,F0,35,E0,02,F0, 2289               <134>
1034 DATA 5E,E0,03,F0,03,4C,DF,17,4C,60,18
     ,AE,F4,03,20,70,17,8E,F4,03, 2059               <057>
1035 DATA BD,CE,1D,AC,F0,03,99,3C,03,A9,1F
     ,99,64,03,AE,F8,03,20,7A,17, 2113               <177>
1036 DATA 8E,F8,03,BD,67,1D,18,90,25,EA,AE
     ,F1,03,AC,F0,03,20,70,17,8E, 2295               <021>
1037 DATA F1,03,BD,6A,1D,99,3C,03,BD,7E,1D
     ,99,64,03,AE,F5,03,20,7A,17, 1983               <045>
1038 DATA 8E,F5,03,BD,5E,1D,99,8C,03,4C,84
     ,17,EA,EA,EA,AE,F2,03,AC,F0, 2762               <034>
1039 DATA 03,20,70,17,8E,F2,03,BD,92,1D,99
     ,3C,03,A9,1E,99,64,03,AE,F6, 2012               <181>
1040 DATA 03,20,7A,17,8E,F6,03,BD,61,1D,99
     ,8C,03,4C,84,17,EA,EA,EA,EA, 2349               <245>
1041 DATA AE,F3,03,AC,F0,03,20,70,17,8E,F3
     ,03,BD,A6,1D,99,3C,03,BD,BA, 2365               <076>
1042 DATA 1D,99,64,03,AE,F7,03,20,7A,17,8E
     ,F7,03,BD,64,1D,99,8C,03,4C, 1968               <058>
1043 DATA 84,17,AC,F0,03,10,01,60,EA,EE,E2
     ,03,CC,E2,03,30,01,60,CE,E2, 2394               <115>
1044 DATA 03,AD,E2,03,C9,27,30,02,60,EA,AE
     ,FA,03,4C,C3,17,8B,17,AD,98, 2233               <201>
1045 DATA 17,C9,05,10,03,EE,98,17,A5,FE,C9
     ,03,10,01,60,C6,00,C6,00,60, 1889               <102>
1046 DATA AD,DF,03,F0,08,A9,00,8D,4B,1D,4C
     ,91,17,4C,AF,17,EA,EA,EA,EA, 2515               <123>
1047 DATA EE,4A,1D,A9,17,CD,4A,1D,F0,03,4C
     ,0C,16,A9,00,8D,4A,1D,8D,0B, 1759               <222>
1048 DATA 90,8E,F0,03,8D,DF,03,4C,C0,17,EC
     ,E2,03,30,01,60,4C,C0,17,0E, 2102               <114>
1049 DATA A6,00,BD,B4,03,C9,15,D0,03,4C,E0
     ,15,C9,0C,F0,4A,C9,0B,F0,06, 2277               <117>
1050 DATA A8,C8,98,18,90,02,A9,08,9D,B4,03
     ,BD,3C,03,85,FB,BD,64,03,85, 2268               <042>
1051 DATA FC,A0,00,A9,20,91,FB,BD,8C,03,20
     ,37,1B,F0,06,8E,F0,03,4C,C0, 2354               <248>
1052 DATA 17,A6,00,BD,B4,03,A0,00,91,FB,A5
     ,FB,9D,3C,03,A5,FC,9D,64,03, 2430               <175>
1053 DATA 60,EA,EA,EA,EA,EA,EA,EA,EA,EA,A9
     ,00,8D,F9,03,BD,3C,03,85,FB, 3400               <134>
1054 DATA 85,01,BD,64,03,85,02,85,FC,20,1F
     ,1B,A9,E9,20,1C,1B,A9,01,20, 1727               <030>
1055 DATA 1C,1B,A9,01,20,1C,1B,A9,16,20,1C
     ,1B,A9,FE,20,1C,1B,A9,16,20, 1323               <011>
1056 DATA 1C,1B,A9,01,20,1C,1B,A9,01,20,1C
     ,1B,A5,01,85,FB,A5,02,85,FC, 1671               <023>
1057 DATA BD,8C,03,20,DA,1B,BD,8C,03,20,DA
     ,1B,A0,00,B1,FB,C9,20,F0,18, 2303               <214>
1058 DATA C9,08,30,14,EE,F9,03,AC,F9,03,C0
     ,08,10,0A,B9,E6,03,9D,8C,03, 2135               <102>
1059 DATA 18,90,CD,EA,A5,01,85,FB,A5,02,85
     ,FC,A6,00,BD,8C,03,20,37,1B, 2321               <058>
1060 DATA F0,06,20,46,1B,4C,C0,17,20,26,1B
     ,A0,0F,A9,E9,20,F3,1B,A0,10, 1818               <100>
1061 DATA A9,01,20,F3,1B,A0,11,A9,01,20,F3
     ,1B,A0,0D,A9,14,20,F3,1B,A0, 1945               <002>
1062 DATA 0E,A9,02,20,F3,1B,A0,14,A9,16,20
     ,F3,1B,A0,13,A9,FF,20,F3,1B, 2065               <203>
1063 DATA A0,12,A9,FF,20,F3,1B,EA,EA,60,A9
     ,00,EA,EA,EA,60,60,EA,EA,EA, 3233               <095>
1064 DATA AE,DE,03,D0,01,60,AD,E0,03,85,FB
     ,AD,E1,03,85,FC,A5,03,20,DA, 2692               <080>
1065 DATA 1B,A5,FC,30,5F,C9,1E,30,5B,C9,20
     ,10,57,EE,5D,1D,A2,0D,EC,5D, 2157               <210>
1066 DATA 1D,F0,4D,48,A5,FB,48,AD,E0,03,85
     ,FB,AD,E1,03,85,FC,A0,00,B1, 2813               <092>
1067 DATA FB,C9,16,D0,04,A9,20,91,FB,91,FB
     ,68,85,FB,68,85,FC,B1,FB,C9, 3285               <095>
1068 DATA 20,30,1B,AE,0C,90,CA,CA,CA,8E,0C
     ,90,CE,0B,90,A9,16,91,FB,A5, 2454               <095>
1069 DATA FB,8D,E0,03,A5,FC,8D,E1,03,60,A0
     ,00,8C,DE,03,4C,61,17,EA,EA, 2690               <091>
1070 DATA A2,00,8E,0C,90,8E,0B,90,8E,DE,03
     ,AD,E0,03,85,FB,AD,E1,03,85, 2442               <131>
1071 DATA FC,A1,FB,C9,16,F0,01,60,A9,20,81
     ,FB,60,EA,EA,AE,E2,03,BD,3C, 3021               <075>
1072 DATA 03,C5,FB,D0,07,BD,64,03,C5,FC,F0
     ,06,CA,E0,00,D0,ED,E8,8E,F0, 3138               <249>
1073 DATA 03,4C,44,17,EA,EA,EA,EA,20,DA,1B
     ,A5,01,C9,20,F0,01,8A,A2,00, 2323               <068>
1074 DATA 81,FB,98,EA,A8,A5,FC,18,69,78,85
     ,FC,98,A2,00,81,FB,A5,FC,38, 3152               <212>
1075 DATA E9,78,85,FC,60,A4,FE,A2,64,CA,10
     ,FD,88,10,FB,4C,E0,15,EA,EA, 3174               <134>
1076 DATA 20,DA,1B,A9,20,A0,00,91,FB,60,A5
     ,FB,9D,3C,03,A5,FC,9D,64,03, 2443               <004>
1077 DATA A9,0C,18,90,1A,EA,EA,20,DA,1B,A5
     ,FC,30,08,C9,1E,30,04,C9,20, 2109               <108>
1078 DATA 30,06,A5,00,8D,F0,03,60,4C,22,1A
     ,A0,00,91,FB,60,EA,EA,EA,EA, 2423               <060>
1079 DATA A5,C5,C9,21,F0,09,C9,22,F0,1B,C9
     ,23,F0,3D,60,AE,E4,03,E0,00, 2609               <113>
1080 DATA F0,07,CA,8E,E4,03,18,90,1A,A2,07
     ,8E,E4,03,18,90,12,AE,E4,03, 2149               <083>
1081 DATA E0,07,F0,04,E8,18,90,E7,A2,00,8E
     ,E4,03,EA,EA,AD,DC,03,85,FB, 2889               <191>
1082 DATA AD,DD,03,85,FC,A2,00,AD,E4,03,81
     ,FB,60,EA,EA,AE,DE,03,E0,00, 2915               <041>
```

Listing 1. Ladeprogramm für das Spiel »Asteroids«. (Fortsetzung)

```
1083 DATA F0,01,60,E8,8E,DE,03,AE,DC,03,8E
     ,E0,03,AE,DD,03,8E,E1,03,AE, 2644      <241>
1084 DATA E4,03,BD,E6,03,85,03,A9,F5,8D,0C
     ,90,A9,F2,8D,0B,90,A9,00,8D, 2517      <252>
1085 DATA 5D,1D,60,CE,CC,C1,CC,C4,8C,DD,30
     ,0A,18,65,FB,85,FB,90,02,E6, 2776      <130>
1086 DATA FC,60,18,65,FB,85,FB,B0,02,C6,FC
     ,60,EA,EA,EA,20,DA,1B,98,A0, 3123      <207>
1087 DATA 00,91,FB,60,32,0B,73,17,00,10,28
     ,44,82,82,FE,82,00,1E,22,42, 1589      <115>
1088 DATA A2,14,08,10,78,24,22,21,22,24,78
     ,00,10,08,14,A2,42,22,1E,00, 955       <241>
1089 DATA 41,7F,41,41,22,14,08,00,08,10,28
     ,45,42,44,78,00,00,0F,12,22, 838       <214>
1090 DATA 42,22,12,0F,00,78,44,42,45,28,10
     ,08,20,50,88,87,41,22,44,38, 1126      <206>
1091 DATA 0C,52,A1,82,84,48,28,18,1C,22,44
     ,82,E1,11,0A,04,18,14,12,21, 1264      <204>
1092 DATA 41,85,4A,30,C0,1F,22,24,12,09,90
     ,49,48,25,14,22,42,44,28,48, 1266      <169>
1093 DATA 3C,02,E1,21,12,21,C1,06,00,03,07
     ,04,08,31,46,88,70,93,0E,04, 1124      <082>
1094 DATA 60,90,09,64,E0,10,30,C0,80,80,40
     ,80,44,64,12,0A,05,02,02,01, 1483      <254>
1095 DATA 04,09,33,C4,18,20,40,80,08,10,10
     ,A0,40,00,00,00,18,18,24,3C, 916       <092>
1096 DATA 42,FF,E7,7E,00,00,00,30,30,00,00
     ,00,E0,CC,0C,20,80,84,31,10, 1571      <020>
1097 DATA 01,00,64,60,08,02,8E,38,50,48,84
     ,08,48,48,24,08,08,40,60,90, 1197      <105>
1098 DATA 60,0C,04,00,42,2C,41,35,2C,46,43
     ,2C,33,30,2C,35,46,2C,43,39, 999       <044>
```


Listing 1. Ladeprogramm für das Spiel »Asteroids«. (Schluß)

```
10 REM ************************            <063>
15 REM                         *           <158>
20 REM ASTEROIDS MAIN PROGRAM *            <182>
25 REM                         *           <168>
30 REM                         *           <173>
31 REM                                     <093>
32 REM 1985 BY CLAUS NEUPERT   *           <251>
33 REM HUMBOLDTSTR. 44         *           <081>
34 REM 4600 DORTMUND 1          *           <055>
35 REM                         *           <178>
36 REM                         *           <179>
37 REM ************************            <090>
40 REM                                     <102>
50 REM POKE 55,40:POKE 56,21               <110>
60 REM  POKE 648,30:SYS 58648              <020>
65 GOSUB 15000                             <125>
70 GOSUB 10000                             <050>
80 GOSUB 11000                             <076>
90 GOSUB 12000                             <102>
200 POKE 6040,1:POKE 6027,10:POKE 7514,0:P
    OKE 7515,0                             <209>
300 S1=36876:S2=36875:S3=36874:SN=36877:SL
    =36878:C=30720                         <010>
310 RA=5:HS=0:HT=0                         <041>
400 POKE 994,38:FOR I=1 TO 38:POKE 1008,I:
    SYS 6080:NEXT:POKE 254,33:POKE 994,5   <078>
500 POKE 36878,15:POKE 36879,8:POKE 36869,
    255:PRINT"{CLR,WHITE}":POKE 988,253:PO
    KE 989,30:HS=1                         <058>
505 POKE 7510,0:POKE 7512,0:POKE 990,0:POK
    E 991,0:POKE 996,0                     <071>
506 POKE 7489,32:POKE 197,64               <183>
510 POKE PEEK(988)+256*PEEK(989),PEEK(996)
    :                                      <162>
520 SYS 5500                               <148>
530 IF PEEK(7510)=3 THEN 700               <016>
540 IF PEEK(7510)=2 THEN 800               <051>
550 IF PEEK(7510)=32 THEN POKE 7510,0:GOTO
     1200                                  <133>
700 POKE S1,0:POKE S2,0:POKE S3,0:M=PEEK(9
    88)+256*PEEK(989)                      <121>
703 POKE SL,15:POKE M,25:POKE M+C,7:POKE S
    N,128:::::FOR Q=1 TO 20:               <072>
704 POKE SL,15-INT(Q/1.333):::::           <242>
705 FOR O=1 TO 16:IF O/2=INT(O/2)THEN POKE
     M+C,5:GOTO 710                        <095>
706 POKE M+C,2                             <196>
710 NEXT:NEXT:POKE SN,0:POKE SL,15:POKE M+
    C,1:GOSUB 2000:POKE 988,253:POKE 989,3
    0                                      <115>
717 IF RA>8 THEN RA=10                     <230>
720 RA=RA-1:IF RA=0 THEN 1500              <075>
725 FOR I=1 TO RA:POKE 7704+I,0:NEXT I:FOR
     J=1 TO 800:NEXT J                     <164>
730 X=PEEK(994):POKE 994,38:FOR I=1 TO 38:
    POKE 1008,I:SYS 6080:NEXT:POKE 994,X:G
    OTO 500                                <042>
800 POKE S1,0:POKE S2,0:POKE S3,0:PO=PEEK(
    828+PEEK(991))+256*PEEK(868+PEEK(991)) <180>
805 POKE PO,24:POKE PO+C,7:POKE SL,15:POKE
     SN,200:FOR L=15 TO 0 STEP-1           <217>
810 POKE SL,L:POKE SN,150+3*L:::::         <181>
820 FOR LX=1 TO 15:IF LX/2=INT(LX/2)THEN P
    OKE PO+C,4:POKE PO,24:GOTO 830         <042>
825 POKE PO+C,2:POKE PO,26                 <244>
830 NEXT LX:NEXT L:POKE SN,0:POKE SL,15:PO
    KE PO,32:POKE PO+C,1                   <019>
840 POKE 1008,PEEK(991):POKE 991,0:SYS 608
    0:SC=SC+250:GOTO 505                   <154>
1200 PF=1:HS=HS+1:HT=1:POKE 7512,0:IF HS/2
     =INT(HS/2)THEN 1250                   <129>
1202 FOR N=7703 TO 7745:IF PEEK(N)=32 THEN
      1210                                 <211>
1205 NEXT                                  <199>
1210 PO=PEEK(988)+256*PEEK(989):POKE PO,32
     :PO=N:IF RND(1)<.30 THEN HS=0:GOTO 70
     0                                     <225>
1215 POKE PO,PEEK(996):PH=INT(PO/256):PL=P
     O-PH*256                              <105>
1216 POKE 988,PL:POKE 989,PH:GOTO 505      <164>
1250 PO=PEEK(988)+256*PEEK(989):POKE PO,32
     :PO=253+256*30:POKE 988,253:POKE 989,
     30                                    <097>
1260 IF PEEK(PO)=32 AND RND(1)<.75 THEN 12
     15                                    <181>
1270 GOTO 700                              <230>
1500 POKE 36869,240:PRINT"{CLR,DOWN,3RIGHT
     ,RED}****************":PRINT TAB(3)"*
     "TAB(18)"*"                           <033>
1502 PRINT TAB(3)"*"TAB(18)"*":PRINT TAB(3
     )"*"TAB(18)"*"                        <225>
1504 PRINT"{3RIGHT}****************"       <039>
1505 PRINT"{HOME,3DOWN,7RIGHT,WHITE}GAME O
     VER"                                  <243>
1510 GOSUB 4000                            <220>
1550 PRINT"{5DOWN,YELLOW}>> SCORE: <<{SPAC
     E,WHITE}"PEEK(7514)+256*PEEK(7515)+SC <020>
1580 GOSUB 2000                            <020>
1590 PRINT"{CYAN,3DOWN,4RIGHT,SPACE,RIGHT}
     AGAIN (Y/N) ?"                        <210>
1600 GET A$:IF A$=""THEN 1600              <126>
1610 IF A$="Y"THEN 90                      <048>
1620 IF A$="N"THEN END                     <082>
1630 GOTO 1600                             <142>
2000 RETURN                                <024>
3000 REM                                   <012>
3010 IF I=1200 THEN POKE 36874,0:POKE 3687
     6,0:GOTO 3110                         <015>
3020 K=K+Z:IF K=0 OR K=250 THEN Z=-Z       <064>
3030 K1=128 OR K2:K2=160 OR K:K3=128 OR K/
     2:K4=180 OR K3:K5=170 OR K1           <196>
3040 IF I<256 OR I>1200 THEN 3110          <202>
3050 IF I=1024 THEN GOSUB 3300:POKE 36878,
     10                                    <133>
3100 POKE 36874,K1:POKE 36876,K3:POKE 3687
     4,K2:POKE 36876,K2:POKE 36874,K3:POKE
      36876,K1                             <011>
3110 POKE 36875,K4:POKE 36875,K5           <239>
3120 RETURN                                <130>
3300 RETURN                                <054>
4000 POKE 36878,15:FOR I=250 TO 130 STEP-2 <155>
4010 POKE 36876,I:POKE 36875,128 OR I:POKE
      36874,240 AND I:NEXT:POKE 36876,0    <198>
4020 POKE 36875,0:POKE 36874,0:POKE 36878,
     15:RETURN                             <046>
```

Listing 2. Hauptprogramm zu »Asteroids«.
Bitte beachten Sie die Eingabehinweise auf Seite 76

```
4999 STOP                                  <239>
10000 PRINT"{CLR}":POKE 36879,8:POKE 36869
      ,240:POKE 36878,15:Z=1               <115>
10005 C$="{WHITE,RED,CYAN,PURPLE,GREEN,BLU
      E,YELLOW}"                           <196>
10010 FOR I=1 TO 20:PRINT MID$(C$,INT(RND(
      1)*7)+1,1);"*";:NEXT                 <243>
10020 PRINT:PRINT:PRINT:PRINT"{WHITE,RIGHT
      }A S T E R O I D S"                  <202>
10030 PRINT:PRINT:PRINT                    <249>
10040 FOR I=1 TO 20:PRINT MID$(C$,INT(RND(
      1)*7)+1,1);"*";:NEXT:PRINT           <190>
10050 PRINT"{5DOWN,WHITE,3RIGHT}BY C. NEUP
      ERT"                                 <163>
10060 PRINT"{4DOWN,3RIGHT,CYAN,3RIGHT}HIT
      ANY KEY"                             <059>
10070 GET A$:IF A$=""THEN GOSUB 3000:GOTO
      10070                                <036>
10080 POKE 36876,0:POKE 36875,0:POKE 36874
      ,0:RETURN                            <179>
11000 PRINT"{CLR,DOWN,3RIGHT,RED}**ASTEROI
      DS**"                                <194>
11010 PRINT"{2DOWN,WHITE}Z{5SPACE}- ROTATE
       LEFT"                               <104>
11020 PRINT"{DOWN}C{5SPACE}- ROTATE RIGHT" <111>
11030 PRINT"{DOWN}B{5SPACE}- FIRE":PRINT"{
      2DOWN,RVSON}SPACE{RVOFF,SPACE}- HYPE
      RSPACE"                              <232>
11040 PRINT"{DOWN,RVSON}F7{RVOFF,4SPACE}-
      THRUST"                              <218>
11050 PRINT"{4DOWN,4RIGHT,CYAN}HIT ANY KEY
      "                                    <202>
11060 GET A$:IF A$=""THEN 11060            <225>
11070 RETURN                               <206>
12000 PRINT"{CLR,DOWN,WHITE}CHOOSE:":PRINT
      :PRINT:PRINT                         <202>
12010 PRINT"1 - SLOW":PRINT:PRINT"2 - MEDI
      UM"                                  <145>
12020 PRINT:PRINT"3 - FAST"                <159>
12030 GET A$:IF A$=""OR VAL(A$)=0 OR VAL(A
      $)>3 THEN 12030                      <094>
12040 K=(4-VAL(A$))*70:POKE 5558,K:RETURN  <210>
15000 RESTORE:FOR I=0 TO 59:READ N:POKE 68
      0+I,N:NEXT                           <119>
15010 FOR I=0 TO 19                        <145>
15020 O=7680+INT(RND(1)*18)+2              <232>
15030 U=8164+INT(RND(1)*18)+2              <251>
15040 L=7724+INT(RND(1)*19)*22             <232>
15050 R=7745+INT(RND(1)*19)*22             <011>
15070 LH=INT(L/256):LL=(L/256-LH)*256      <132>
15080 OH=INT(O/256):OL=(O/256-OH)*256      <234>
15090 UH=INT(U/256):UL=(U/256-UH)*256      <175>
15095 RH=INT(R/256):RL=(R/256-RH)*256      <087>
15100 POKE 7530+I,LL:POKE 7550+I,LH:POKE 7
      570+I,OL                             <013>
15110 POKE 7590+I,RL:POKE 7610+I,RH: POKE
      7630+I,UL                            <026>
15120 NEXT I                               <218>
15130 FOR I=0 TO 2:READ NU,NL,NO,NR        <102>
15140 POKE 7527+I,NU:POKE 7518+I,NL: POKE
      7521+I,NO:POKE 7524+I,NR: NEXT I     <211>
15150 FOR I=0 TO 7:POKE 1009+I,0:NEXT: POK
      E 1018,0:POKE 1019,0                 <047>
15160 FOR I=0 TO 8:READ N:POKE 997+I,N:NEX
      T I                                  <243>
15170 FOR I=0 TO 9: READ N: POKE 7500+I,N:
      NEXT I                               <143>
15200 POKE 7489,32:POKE 7512,0:POKE 1008,1
      :POKE 7490,0:POKE 7499,0:POKE 7498,0 <136>
15300 FOR I=7168+32*8 TO 7168+32*8+7:POKE
      I,0:NEXT I                           <037>
15900 RETURN                               <210>
20000 DATA 24,109,90,29,141,90,29,176,1,96
      ,238,91,29,165,254                   <189>
20010 DATA 201,3,48,4,198,254,198,254,173,
      226,3,201,35,16,8                    <103>
20020 DATA 201,35,240,4,238,226,3,234,173,
      152,23,201,5,240,13                  <135>
20030 DATA 201,5,16,9,238,152,23,234,234,2
      34,234,234,234,96,96                 <006>
20040 REM                                  <034>
20050 DATA 235,235,21,255,234,1,22,233,233
      ,23,23,21                            <221>
20060 DATA 0,234,235,1,23,22,21,255,233    <250>
20070 DATA 0,1,255,23,22,21,235,234,233,0  <223>
```

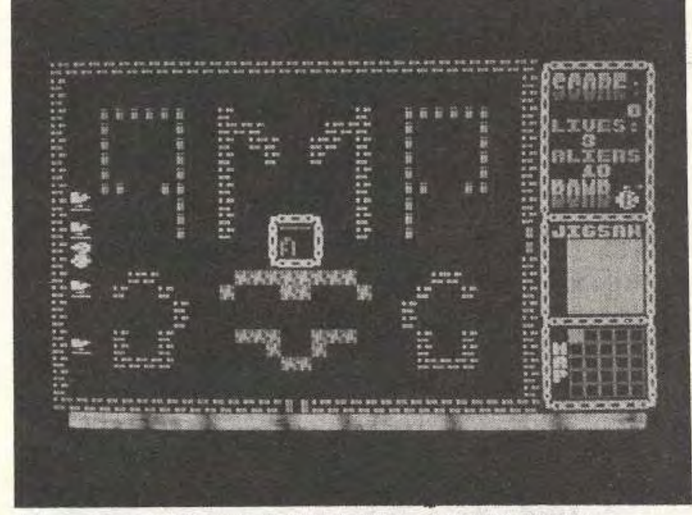

Listing 2. Hauptprogramm zum Spiel »Asteroids«. (Schluß)

C16–Spieleführer

Mit etwas Verspätung haben die Softwarefirmen den C16 entdeckt und in den letzten Monaten eine ganze Reihe von Spielen veröffentlicht. Wir haben Ihnen eine Übersicht mit vielen Kauf-Tips zusammengestellt.

Wenn Sie Ihren C16/116 zum Spiel-Computer verwandeln wollen, dann haben wir hier einige interessante Anregungen für Sie parat. Zunächst ein paar Worte zum lieben Geld: Die meisten Spiele-Kassetten kosten zwischen 25 und 39 Mark, was im Vergleich zu anderen Software-Preisen recht human ist. Einige Firmen bieten aber auch Preisbrecher für 9,95 Mark an, die oft besser sind als teurere Titel. Vor allem das Angebot von Mastertronics ist ebenso preiswert wie qualitativ hochwertig. Alle »Billigspiele« dieses Anbieters haben wir in der Übersicht mit einem Sternchen gekennzeichnet. Unsere Übersicht erhebt übrigens keinen Anspruch auf Vollständigkeit.

Dann mal ran an die Joysticks. Es gibt einige Spiele-Sammlungen für den C16, die mehr Software-Spaß fürs gleiche Geld bieten. Von Gremlin Graphics gibt es die Zusammenstellung »C16's Classics«, die aus »Dork's Dilemma«, »Tycoon Tex« (beides Geschicklichkeits-Spiele), »Xargon Wars« und »Petals of Doom« (zwei Schießspiele) besteht. Trotz kleiner Schwächen bei einzelnen Titeln eine empfehlenswerte Zusammenstellung, bei der man vier professionell gemachte, schnelle Spiele für sein Geld bekommt.

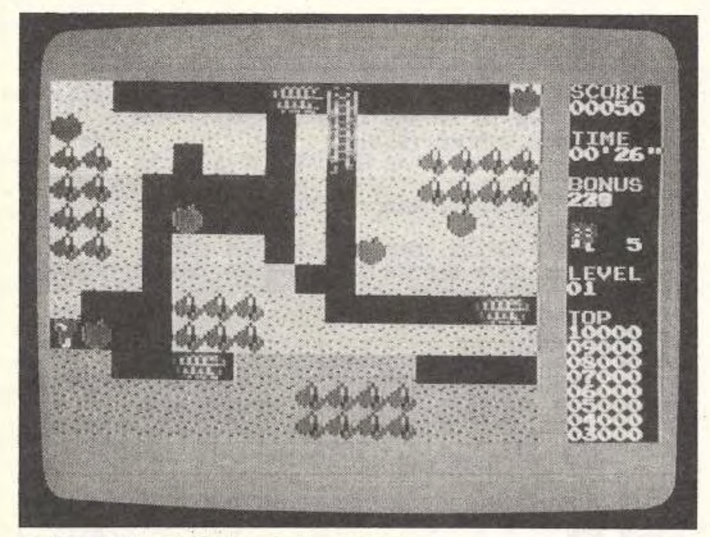


Eines von vier Spielen der Zusammenstellung »C16's Classics«

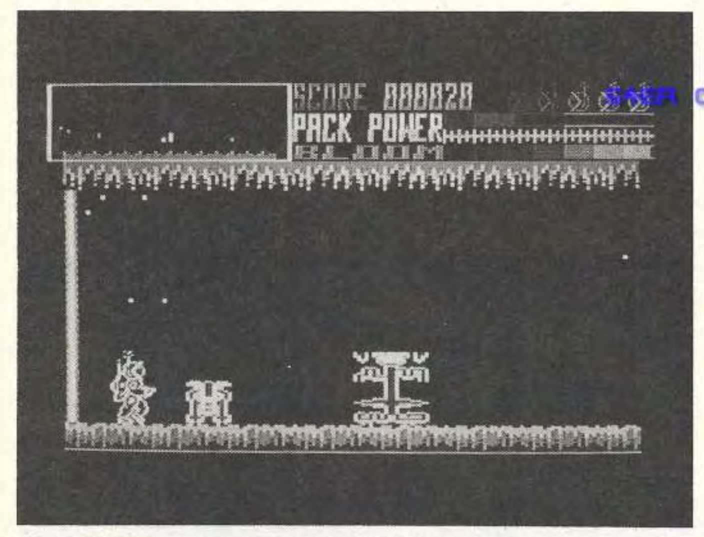
Ebenfalls vier Titel findet man auf Anirogs »Favourite Four«-Kassette. Es handelt sich hier um »Flight Path 737« (Flugsimulation), »Moon Buggy« (Schießspiel), »Las Vegas« (Spiel-

automat) und »Zodiac« (Geschicklichkeit) – eine abwechslungsreiche Mischung.

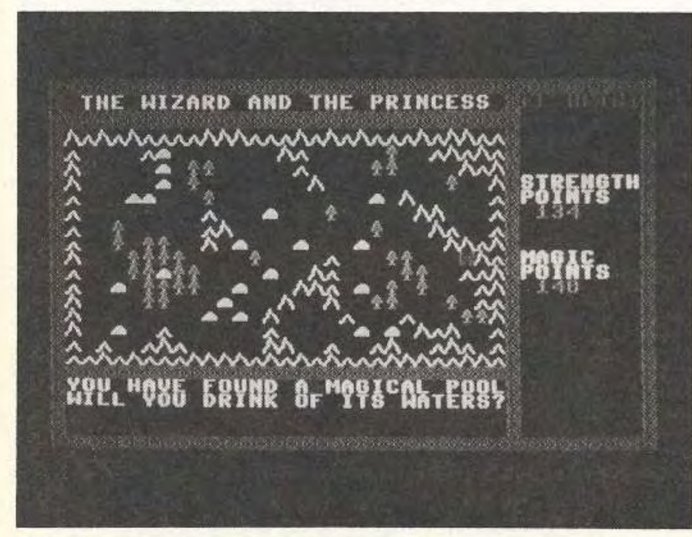
Gleich 15 Spiele findet man auf dem »C 16 Games Pack I« von Melbourne House. Allerdings sind es recht simple Programme, die alle in Basic geschrieben sind. Das hat aber auch den Vorteil, daß man sie LISTen und beliebig verändern kann. Ein lehrreiches Vergnügen für Basic-Einsteiger und Fortgeschrittene. Vom Karten- bis zum Wirtschafts-Spiel ist hier fast die gesamte Palette vertreten.



»Tutti-Frutti«, ein tolles 10-Mark-Spiel



Ein Baller-Programm aus den »C16's Classics«



Äußerst abenteuerlich: «Wizard and the Princess«

Die Spiele-Zukunft für den C 16 sieht gar nicht mal übel aus. Vor ein paar Wochen hat U.S. Gold, eine britische Mega-Softwarefirma, mit dem Baller-Evergreen »Beach Head« erstmals einen ihrer Erfolgstitel für den »Kleinen« umgesetzt. Vom Erfolg dieses Experiments wird es wohl abhängen, ob weitere C 16-Adaptionen folgen werden. Andere große englische Softwarehäuser wie Ocean und Melbourne House haben beim C 16-Software-Geschäft bereits ihre Finger im Spiel. Man darf gespannt sein.

Sportspiel-Nachschub

Zu Redaktionsschluß war ein sehr interessantes Projekt in Arbeit, das in den nächsten Wochen auf den Markt kommen soll: »Winter-Olympiade« von Kingsoft. Das Spiel soll das etwas magere Angebot an Sport-Simulationen für den C 16 stark bereichern und hat sich den C 64-Hit »Winter Games« als Vorbild genommen. Grafik und Sound können natürlich nicht so gut werden wie beim großen Computer-Bruder, aber für C 16-Verhältnisse soll »Winter-Olympiade« Maßstäbe setzen. Unter den sechs bis acht Disziplinen sollen sich Ski-Slalom und Biathlon befinden. Ein eiskaltes Vergnügen rechtzeitig zum Frühlings-Beginn. (hl)

Titel	Spiele-Typ	Anbieter
3D Time Trek	Schießspiel	J, K, R
Ace	Flugsimulation	J, K, R
Beach Head	Schießspiel	J, K, R
Berks	Schießspiel	J, K, R
Berks 3	Schießspiel	J, K, R
BMX Racers *	Radrennen	M, J
Bongo Construction Set	Geschicklichkeit	K
C 16's Classics	Spiele-Sammlung	J, K, R
C 16 Games Pack 1	Spiele-Sammlung	J, K, R
Canoe Slalom	Kanu-Slalom	J, K, R
Cave Fighter	Geschicklichkeit	J, K, R
Dark Tower	Geschicklichkeit	J, K, R
Defence 16	Schießspiel	J, K, R
Favourite Four	Spiele-Sammlung	J, K, R
Formula 1 Simulator *	Autorennen	M, J
Galaxy	Schießspiel	K
Ghost Town	Geschicklichkeit	K
Grandmaster	Schach	K
Hustler	Billard	J, K, R
Invasion 2000 A.D.	Schießspiel	J, K, R
Jump Jet	Flugsimulation	K
Minipedes	Schießspiel	J, K, R
Out on a Limb	Geschicklichkeit	J, K, R
Petch	Geschicklichkeit	J, K, R
Robin to the Rescue	Geschicklichkeit	J, K, R
Rockman *	Geschicklichkeit	M, J
Roller Kong	Geschicklichkeit	J, K, R
Scramble	Schießspiel	J, K, R
Spectipede *	Schießspiel	M, J
Squirm *	Geschicklichkeit	J, K, R
Star Commander	Schießspiel	J, K, R
Thompson's Star Events	Sportspiel	J, K, R
Tom	Geschicklichkeit	K
Tutti Frutti *	Geschicklichkeit	M, J
Vegas Jackpot *	Spielautomat	M, J
Torpedo Run	Schießspiel	J, K, R
Winter-Olympiade	Sportspiel	K
Wizard & the Princess	Geschicklichkeit/Strategie	J, K, R
World Cup Football	Fußball	J, K, R

Die Programme und nähere Informationen gibt's bei folgenden Firmen:
R = Rushware, An der Gümpgesbrücke 24, 4044 Kaarst 2
K = Kingsoft, Schnackebusch 4, 5106 Roetgen, Tel. (02408) 5119
M = Mastertronic GmbH, Kaiser-Otto-Weg 18, 4770 Soest, Tel. (02921) 75028
J = Joysoft, Humboldtstr. 84, 4000 Düsseldorf 1, Tel. (0211) 6801403
* = »Billigspiel« für 9,90 Mark

Auf der Rennstrecke

Das erste Autorennen für den C16 ist da. »Turbo-Racer«, ein Spielprogramm mit Gangschaltung.

Turbo-Racer« ist vollständig in Maschinensprache geschrieben und natürlich entsprechend flott. Wenn Sie das Programm (Listings 1 und 2) mit Hilfe des eingebauten Maschinensprachemonitors eingetippt haben, speichern Sie es am besten sofort (eingeben: `S "TURBO-RACER",1,2000,3E68` für Datasette; `S "TURBO-RACER",8,2000,3E68` für Floppy). Das hat auch seinen Grund, denn schon beim geringsten Tippfehler kann das Programm in der Speicherwüste Ihres Computers verschwinden, oder das vorhandene Programm wird gar zerstört. In der Tabelle 1 haben wir Ihnen die einzelnen Routinen des Programms aufgelistet.

Starten Sie »Turbo-Racer« mit SYS 9506, und ein Menü empfängt Sie, um Ihnen die freie Wahl zwischen Tastatur und Joystick zu lassen. Nachdem Sie der Computer darüber in Kenntnis gesetzt hat, wie die Gänge zu schalten sind (Tabelle 2), können Sie durch einen beliebigen Tastendruck mit dem Spiel beginnen.

Joystick:	Gang 1:	Feuer + Hebel nach hinten
	Gang 2:	Feuer
	Gang 3:	Feuer + Hebel nach vorne
Tastatur:	Gang 1:	A-Taste
	Gang 2:	S-Taste
	Gang 3:	D-Taste
Steuerung:	< = links	
	> = rechts	

Tabelle 2. Bedienung von »Turbo-Racer«

Ziel des Spiels ist es, mit Ihrem Fahrzeug so weit wie möglich zu kommen, ohne ein entgegenkommendes Auto oder die Leitplanke zu rammen. Mit der Zeit wird es immer schwieriger, diese Aufgabe zu bewältigen, da die Fahrbahn immer schmaler wird, die Hindernisse zunehmen und die Spielgeschwindigkeit steigt. Zuletzt sind nur noch wenige Hindernisse auf dem Bildschirm, denn die Fahrbahn ist gerade noch so breit, daß Ihr Fahrzeug hindurchfahren kann. Aber probieren Sie es doch selbst.

(Eduard Chen/kn)

ADRESSE	VARIABLE/ROUTINE
$2000-$2048	JOY/KEY-Abfrage + CRASH CHECK
$2049-$209F	Hindernisse erzeugen + SCORE und neue Parameter setzen
$20A0-$20E3	Initialisierung
$20E4-$2139	Fahrbahn erzeugen
$201C-$202E	Spielfigur löschen
$2034-$203C	Spielfigur POKEn
$2247-$225F	Zufallsgenerator/Zahl
$2260	Zahl von Zufallsgenerator
$2267-$2275	Filtert Werte von Zufallsgenerator aus. Zufallszahl muß <=($2266) sein
$2266	Grenze für Zufallszahlen
$2191-$21CE	Routine zum SCORE setzen
$24FB-$2521	JOYSTICK-Abfrage von PORT 1/Wert in Y-Register
$24C7-$24F8	KEYBOARD-Abfrage und Ausführung
$2522-$2590	Anfangsbild PRINTen + KEY/JOY OPTION wählen
$2441-$2462	Neuen Zeichensatz einschalten und zu Spielstart springen
$22E5	Beginn der Hauptschleife (Spielanfang)
$DA5E	SCROLL-Routine (aus Betriebssystem)
$D88D	CLEARSCREEN-Routine
$21FF	Position von Spielfigur (1 bis 40)
$21FE	CRASH-FLAG
$21FC	Anzahl der Hindernisse/Zeile
$21F4	Fahrbahnbreite
$21F3	FLAG für Richtung der Spielfigur (LI/MI/RE)
$21F2	Kurvenhäufigkeit
$2200/$2201	SCORE/LOW-HIGH
$220E	Wert für Verzögerung/geringer Wert – hohe Ablaufgeschwindigkeit
$C6	KEYBOARD-Buffer
$E3CE	Fragt STOP-Taste ab
$EBD9	GET-Routine/ASCII-Wert in Akku
$2385-$23EF	Tonerzeugen (Kopie von Betriebssystem)
$B8E	Lautstärke setzen/davor Lautstärke in X-Register laden
$03/$04	Tonlänge (LOW-HIGH)
$3BE3	SOUND-Generator (1 bis 3)
$3BE4/$3BE5	Tonhöhe (LOW-HIGH)
$FFD2	Zeichenausgabe/ASCII-Wert in Akku
$0C00	Start des Bildschirmspeichers (Text)

Tabelle 1. Routinen für »Turbo-Racer«

```
>2000 AE FF 21 AD FD 3F C9 01
>2008 F0 03 4C 0D 24 4C 82 24
>2010 A9 20 9D C0 0F CA 10 01
>2018 E8 4C 27 20 A9 20 9D C0
>2020 0F E8 E0 28 D0 01 CA BD
>2028 C0 0F C9 20 F0 06 4C 3D
>2030 20 EA EA 60 A9 00 9D C0
>2038 0F 8E FF 21 60 4C 82 23
>2040 60 EA EA EA EA EA EA EA
>2048 EA A2 00 A9 02 AC FB 21
>2050 E8 99 28 0C 98 69 07 A8
>2058 A9 02 EC FC 21 D0 F1 AC
>2060 01 22 AE 00 22 E8 8E 00
>2068 22 E0 FF D0 04 C8 8C 01
>2070 22 C0 00 D0 04 E0 C8 F0
>2078 19 C0 01 D0 04 E0 90 F0
>2080 14 C0 03 D0 04 E0 20 F0
>2088 0F C0 04 D0 04 E0 B0 F0
>2090 0A 60 4C 4F 21 4C 5C 21
>2098 4C 69 21 4C 76 21 EA EA
>20A0 A9 07 8D 08 22 A9 11 8D
>20A8 09 22 A9 00 8D FE 21 A9
>20B0 14 8D FF 21 A9 01 8D FC
>20B8 21 A9 00 8D 00 22 A9 28
>20C0 38 ED 09 22 18 69 01 8D
>20C8 0B 22 A9 02 8D 0A 22 A9
>20D0 90 8D 04 22 A9 01 8D 05
>20D8 22 A9 20 8D 06 22 A9 03
>20E0 8D 07 22 60 AD 02 22 38
>20E8 ED 08 22 30 38 AE 03 22
>20F0 AD 0A 22 C9 00 D0 0D E8
>20F8 EC 0B 22 D0 01 CA 8E 03
>2100 22 4C 0D 21 CA E0 00 D0
>2108 01 E8 8E 03 22 A9 01 9D
>2110 27 0C AD 09 22 18 6D 03
>2118 22 8D 0C 22 A9 01 AE 0C
>2120 22 9D 27 0C 60 AD 0A 22
>2128 C9 00 D0 08 A9 01 8D 0A
>2130 22 4C ED 20 A9 00 8D 0A
>2138 22 4C ED 20 A5 C6 C9 38
>2140 D0 09 A9 14 A2 08 A0 01
>2148 4C 7F 21 C9 3B D0 09 A9
>2150 0F A2 0B A0 02 4C 7F 21
>2158 C9 08 D0 09 A9 0B A2 0F
>2160 A0 02 4C 7F 21 C9 0B D0
>2168 09 A9 08 A2 14 A0 03 4C
>2170 7F 21 C9 10 D0 C6 A9 04
>2178 A2 1B A0 01 8C 0E 22 8D
>2180 09 22 8E 08 22 8C FC 21
>2188 A9 29 38 ED 09 22 8D 0B
>2190 22 AD 09 22 C9 0F D0 07
>2198 A0 00 A2 C8 4C C8 21 C9
>21A0 0B D0 07 A2 90 A0 01 4C
>21A8 C8 21 C9 08 D0 07 A2 20
>21B0 A0 03 4C C8 21 C9 04 D0
>21B8 07 A0 04 A2 B0 4C C8 21
>21C0 C9 14 D0 0A A2 00 A0 00
>21C8 8E 00 22 8C 01 22 AD 01
>21D0 22 60 EA 8D 0B 22 60 AD
>21D8 60 22 8D 03 22 20 67 22
```

Listing 1. »Turbo-Racer« (Teil 1). Bitte mit eingebautem Monitor eingeben.

```
>21E0 AD 60 22 8D FB 21 A9 64
>21E8 8D 66 22 20 67 22 AD 60
>21F0 22 8D 02 22 20 49 20 20
>21F8 5E DA EA CB 01 EA 00 1A
>2200 26 00 10 0E 90 01 20 03
>2208 07 11 00 18 1F EA EA EA
>2210 AD 01 22 C9 05 F0 01 60
>2218 A9 14 CD 00 22 D0 F8 A2
>2220 00 A9 01 EC 03 22 F0 07
>2228 9D 27 0C E8 4C 23 22 A2
>2230 28 EC 0C 22 F0 07 9D 27
>2238 0C CA 4C 31 22 A9 FF 8D
>2240 FE 21 60 EA EA EA EA 38
>2248 AD 61 22 6D 64 22 6D 65
>2250 22 8D 60 22 A2 04 BD 60
>2258 22 9D 61 22 CA 10 F7 60
>2260 10 10 CB 5B F9 78 64 20
>2268 47 22 AD 60 22 18 69 01
>2270 ED 66 22 10 F2 60 EA EA
>2278 EA A9 27 8D 66 22 20 67
>2280 22 AD 60 22 8D 03 22 20
>2288 67 22 AD 60 22 8D FB 21
>2290 A9 64 8D 66 22 20 67 22
>2298 AD 60 22 8D FA 22 20 49
>22A0 20 20 5E DA 20 00 20 20
>22A8 E4 20 20 10 22 A9 27 8D
>22B0 66 22 A2 20 A0 FF 88 C0
>22B8 00 D0 FB E0 00 CA D0 F4
>22C0 4C 87 22 A9 0A 8D 66 22
>22C8 20 67 22 AD 60 00 8D CD
>22D0 22 EA EA EA EA EA EA A9
>22D8 16 8D 66 22 20 67 22 AD
>22E0 60 22 8D 03 22 20 67 22
>22E8 AD 60 22 8D FB 21 A9 64
>22F0 8D 66 22 20 67 22 AD 60
>22F8 22 8D 02 22 20 E4 20 20
>2300 5E DA 20 00 20 20 49 20
>2308 20 10 22 20 62 20 A9 27
>2310 8D 66 22 AE FE 3F A0 FF
>2318 88 C0 00 D0 FB CA E0 00
>2320 D0 F4 AC FE 3F C0 15 D0
>2328 1A 20 00 24 A9 0F A0 F4
>2330 20 84 3B 20 85 23 20 FA
>2338 23 A9 08 A0 00 20 84 3B
>2340 20 85 23 C0 30 D0 1A 20
>2348 00 24 A9 0F A0 A4 20 84
>2350 3B 20 85 23 20 FA 23 A9
>2358 02 A0 30 20 84 3B 20 85
>2360 23 C0 80 D0 1A 20 00 24
>2368 A9 0F A0 F4 20 84 3B 20
>2370 85 23 20 FA 23 A9 02 A0
>2378 20 20 84 3B 20 85 23 4C
>2380 E5 22 4C 55 25 AE E3 3B
>2388 CA E0 03 B0 10 86 80 AC
>2390 E4 3B AD E5 3B 84 7E 85
>2398 7F A4 03 A6 80 A5 04 E0
>23A0 02 D0 01 CA 48 C0 00 D0
>23A8 07 C9 00 D0 03 C8 D0 0F
>23B0 98 48 20 C0 8C BD FE 04
>23B8 1D FC 04 D0 F6 68 A8 98
>23C0 49 FF 18 69 01 EA 9D FC
>23C8 04 68 49 FF 69 00 9D FE
>23D0 04 A5 7E 9D 0E FF BD B8
>23D8 B8 AA BD 10 FF 29 FC 05
>23E0 7F 9D 10 FF A6 80 BD BA
>23E8 B8 0D 11 FF 8D 11 FF 60
>23F0 EA A9 02 85 03 A9 00 85
>23F8 04 60 A9 01 8D E3 3B 60
>2400 A9 03 8D E3 3B 60 8D E5
>2408 3B 8C E4 3B 60 A5 C6 C9
>2410 2F D0 03 4C 10 20 C9 2C
>2418 D0 03 4C 1C 20 C9 0A D0
>2420 08 A9 80 8D FE 3F 4C 27
>2428 20 C9 0D D0 08 A9 30 8D
>2430 FE 3F 4C 27 20 C9 12 D0
>2438 05 A9 15 8D FE 3F 4C 27
>2440 20 20 8B D8 20 A0 20 A9
>2448 00 8D 01 22 A9 80 8D FE
>2450 3F A2 08 20 C0 B8 20 6F
>2458 3B A9 00 85 37 8D 12 FF
>2460 A9 32 85 38 A9 3C 8D 13
>2468 FF 4C E5 22 F8 A9 3C 8D
>2470 13 FF A9 00 8D 12 FF A9
>2478 00 85 33 A9 20 85 34 4C
>2480 E5 22 20 FB 24 AE FF 21
>2488 C0 07 D0 03 4C 10 20 C0
>2490 03 D0 03 4C 1C 20 C0 81
>2498 D0 08 A9 15 8D FE 3F 4C
>24A0 27 20 C0 85 D0 05 A9 80
>24A8 8D FE 3F C0 80 D0 05 A9
>24B0 30 8D FE 3F C0 04 F0 DB
>24B8 C0 06 F0 D0 C0 02 F0 D3
>24C0 C0 08 F0 C8 4C 27 20 A5
>24C8 C6 C9 2F D0 03 4C 10 20
>24D0 C9 2C D0 03 4C 1C 20 C9
>24D8 0A D0 08 A9 80 8D FE 3F
>24E0 4C 27 20 C9 0D D0 08 A9
>24E8 30 8D FE 3F 4C 27 20 C9
>24F0 12 D0 05 A9 15 8D FE 3F
>24F8 4C 27 20 A2 00 BD FB BF
>2500 AA 78 8E 08 FF AD 08 FF
>2508 8E 08 FF CD 08 FF D0 F2
>2510 58 49 FF A8 29 0F AA BD
>2518 F0 BF C0 0F 90 02 09 80
>2520 A8 60 A2 00 8A 48 BD 09
>2528 3D C9 FF F0 09 20 D2 FF
>2530 68 AA E8 4C 24 25 20 D9
>2538 EB C9 00 F0 F9 C9 31 D0
>2540 08 A9 01 8D FD 3F 4C 62
>2548 25 C9 32 D0 E9 A9 00 8D
>2550 FD 3F 4C 6D 25 A9 FF 8D
>2558 12 FF A9 D0 8D 13 FF 4C
>2560 22 25 A9 00 85 30 A9 3E
>2568 85 31 4C 75 25 A9 33 85
>2570 30 A9 3E 85 31 A0 00 B1
>2578 30 C9 FF F0 07 20 D2 FF
>2580 C8 4C 77 25 20 D9 EB C9
>2588 00 F0 F9 A9 10 8D 03 22
>2590 4C 41 24 00 FF FF FF FF
```

Listing 1. »Turbo-Racer« (Teil 1, Schluß)


```
>3B00 00 00 00 00 00 00 00 00
>3B08 00 00 00 00 00 00 00 00
>3B10 00 00 00 00 00 00 00 00
>3B18 00 00 00 00 00 00 00 00
>3B20 01 CA 48 C0 00 D0 07 C9
>3B28 00 D0 03 C8 D0 0F 98 48
>3B30 20 C0 8C BD FE 04 1D FC
>3B38 04 D0 F6 68 A8 98 49 FF
>3B40 18 69 01 EA 9D FC 04 68
>3B48 49 FF 69 00 9D FE 04 A5
>3B50 7E 9D 0E FF BD B8 B8 AA
>3B58 BD 10 FF 29 FC 05 7F 9D
>3B60 10 FF A6 80 BD BA B8 0D
>3B68 11 FF 8D 11 FF EA 60 A9
>3B70 02 85 03 A9 00 85 04 60
>3B78 A9 01 8D E3 3B 60 A9 03
>3B80 8D E3 3B 60 8D E5 3B 8C
>3B88 E4 3B 60 A5 C6 C9 2F D0
>3B90 03 4C 10 20 C9 2C D0 03
>3B98 4C 1C 20 C9 0A D0 08 A9
>3BA0 80 8D FE 3F 4C 27 20 C9
>3BA8 0D D0 08 A9 30 8D FE 3F
>3BB0 4C 27 20 C9 12 D0 05 A9
>3BB8 15 8D FE 3F 4C 27 20 FF
>3BC0 FF FF FF FF FF FF FF FF
>3BC8 FF FF FF FF FF FF FF FF
>3BD0 FF FF FF FF FF FF FF FF
>3BD8 FF FF FF FF FF FF FF FF
>3BE0 FF FF FF 01 20 02 FF FF
>3BE8 FF FF FF FF FF FF FF FF
>3BF0 FF FF FF FF FF FF FF FF
>3BF8 FF FF FF FF FF FF FF FF
>3C00 3C 99 FF 99 18 99 FF C3
>3C08 FF 24 FF 92 FF 24 FF 92
>3C10 3C 42 C3 FF DB FF E7 3C
>3C18 3C 00 00 00 00 00 00 00
>3C20 00 00 00 00 00 00 00 00
>3C28 00 00 00 00 00 00 00 00
>3C30 00 00 00 00 00 00 00 00
>3C38 00 00 00 00 00 00 00 00
>3C40 00 00 00 00 00 00 00 00
>3C48 00 00 00 00 00 00 00 00
>3C50 00 00 00 00 00 00 00 00
>3C58 00 00 00 00 00 00 00 00
>3C60 00 00 00 00 00 00 00 00
>3C68 00 00 00 00 00 00 00 00
>3C70 00 00 00 00 00 00 00 00
>3C78 00 00 00 00 00 00 00 00
>3C80 FF FF FF FF FF FF FF FF
>3C88 FF FF FF FF FF FF FF FF
>3C90 FF FF FF FF FF FF FF FF
>3C98 FF FF FF FF FF FF FF FF
>3CA0 FF FF FF FF FF FF FF FF
>3CA8 FF FF FF FF FF FF FF FF
>3CB0 FF FF FF FF FF FF FF FF
>3CB8 FF FF FF FF FF FF FF FF
>3CC0 FF FF FF FF FF FF FF FF
>3CC8 FF FF FF FF FF FF FF FF
>3CD0 FF FF FF FF FF FF FF FF
>3CD8 FF FF FF FF FF FF FF FF
>3CE0 FF FF FF FF FF FF FF FF
>3CE8 FF FF FF FF FF FF FF FF
>3CF0 FF FF FF FF FF FF FF FF
>3CF8 FF FF FF FF FF FF FF FF
>3D00 00 00 00 00 00 00 00 00
>3D08 00 93 11 12 2A 2A 2A 2A
>3D10 2A 2A 2A 2A 2A 2A 2A 2A
>3D18 2A 20 20 54 55 52 42 4F
>3D20 2D 52 41 43 45 52 20 20
>3D28 2A 2A 2A 2A 2A 2A 2A 2A
>3D30 2A 2A 2A 2A 92 11 20 20
>3D38 20 20 20 20 20 20 20 20
>3D40 20 42 59 20 45 44 57 41
>3D48 52 44 20 43 48 45 4E 20
>3D50 31 39 38 35 20 20 20 20
>3D58 20 20 20 20 20 20 20 20
>3D60 20 20 20 20 20 20 20 20
>3D68 20 20 20 20 20 20 20 20
>3D70 20 20 20 20 20 20 20 20
>3D78 20 20 20 20 20 20 20 20
>3D80 20 20 20 20 20 20 11 11
>3D88 20 20 20 20 20 20 20 20
>3D90 20 20 12 20 42 49 54 54
>3D98 45 20 57 41 45 48 4C 45
>3DA0 4E 20 53 49 45 3A 20 92
>3DA8 11 11 9D 9D 9D 9D 9D 9D
>3DB0 9D 9D 9D 9D 9D 9D 9D 9D
>3DB8 9D 9D 9D 9D 9D 9D 12 20
>3DC0 31 20 92 20 2E 2E 2E 2E
>3DC8 2E 2E 2E 4A 4F 59 53 54
>3DD0 49 43 4B 20 20 20 20 20
>3DD8 20 20 20 20 20 20 20 20
>3DE0 20 20 20 20 20 20 20 20
>3DE8 11 12 20 32 20 92 20 2E
>3DF0 2E 2E 2E 2E 2E 2E 4B 45
>3DF8 59 42 4F 41 52 44 FF 00
>3E00 11 20 20 20 20 20 20 20
>3E08 20 20 20 20 20 20 20 47
>3E10 41 45 4E 47 45 3A 12 46
>3E18 45 55 45 52 92 2F 12 46
>3E20 92 2B 56 4F 52 4E 45 2F
>3E28 12 46 92 2B 48 49 4E 54
>3E30 45 4E FF 11 20 20 20 20
>3E38 20 20 20 20 20 20 20 20
>3E40 12 53 54 45 55 45 52 55
>3E48 4E 47 92 3A 20 3C 20 3E
>3E50 20 2D 54 41 53 54 45 4E
>3E58 2F 12 47 41 45 4E 47 45
>3E60 92 3A 41 2F 53 2F 44 FF
```

Listing 2. »Turbo-Racer« (Teil 2). Bitte mit eingebautem Monitor eingeben.

Das Boot – Ein tolles Spiel für den VC 20

Sie kennen den Spielfilm »Das Boot«? Der Film wird durch dieses Programm simuliert. Lassen Sie sich die ausgezeichneten grafischen und akustischen Effekte nicht entgehen. Spielen Sie U-Boot-Kommandant von U-96.

Als Kommandant von U-96 haben Sie und Ihre Mannschaft die Sinnlosigkeit des Krieges erkannt und wollen zurück in die Heimat. In einem unterirdischen Höhlensystem von Gibraltar müssen Sie den Minen und Felsen ausweichen. Die Aufgabe besteht nun darin, die Proviantpakete einzusammeln, die von Deutschen für Sie ausgelegt sind. Ein Spiel für den VC 20 mit 16 KByte Speichererweiterung.

Erklärung der drei Szenen:

Szene 1: Es ist Tag, es ist nur ein Proviant ausgelegt.
Szene 2: Es ist Nacht, es sind zwei Proviantpakete vorhanden.
Szene 3: Sie tauchen in die Tiefe, es sind fünf Proviantpakete vorhanden.

Erklärung der CHANGE STAGE jeweils nach einer Szene:

Nummer 1: Sie müssen den verrückt gewordenen Johann zur Vernunft bringen.
Nummer 2: Sie müssen entscheiden, wieviel Proviantpakete sie einem zweiten U-Boot abgeben.
Nummer 3: Sie sind zu tief getaucht und müssen an die Oberfläche. Geben Sie Ihren Lösungswunsch 1, 2 oder 3 ein.

Die CHANCE STAGE ist eine Texteinlage, die Abwechslung in das Actionspiel bringt.

Geben Sie vor dem Abtippen im Direktmodus ein:

1) POKE 44,28:POKE 28*256,0:NEW
2) Tippen Sie den »BOOT«-Lader (Listing 1) ab.
3) Speichern Sie diesen.
4) Geben Sie NEW ein.
5) Tippen Sie das Hauptprogramm (Listing 2) ab.
6) Speichern Sie es unter dem Namen »Das Boot«, da der Lader auf diesen Namen programmiert ist.

Wenn Sie nach dem Abtippen nun endlich spielen wollen, ist wie folgt vorzugehen:

1) Geben Sie im Direktmodus ein:
   POKE 44,28:POKE 28*256,0:NEW
2) Laden Sie jetzt den Lader, und wenn der Cursor wieder blinkt, einfach die RETURN-Taste drücken. Das Hauptprogramm wird automatisch nachgeladen.
3) Jetzt brauchen Sie nur noch starten und können, ohne sich zu langweilen, stundenlang spielen.

(C.M. Grzibek/R. Desideri/ah)


```
0 PRINT"{CLR,2DOWN}LOAD"CHR$(34)"DAS BOOT"
  CHR$(34)",8";:PRINT"{HOME}";             <157>
1 FOR I=256 TO 264:POKE 5120+I,0:NEXT      <066>
2 FOR I=0 TO 1463:READ A:POKE 5120+I,A:NEX
  T:NEW                                    <020>
9 DATA 0,0,0,0,0,0,0,0                     <183>
10 DATA 63,63,51,51,63,115,115,115         <194>
11 DATA 62,51,51,60,51,115,127,126         <122>
12 DATA 63,63,51,48,48,115,127,127         <068>
13 DATA 60,62,51,51,51,115,127,127         <061>
14 DATA 63,63,51,60,60,115,127,127         <094>
15 DATA 63,63,48,60,60,112,112,112         <107>
16 DATA 63,63,51,48,55,115,127,127         <197>
17 DATA 51,51,51,63,63,115,115,115         <176>
18 DATA 63,63,28,28,28,124,127,127         <055>
19 DATA 30,30,6,6,102,103,127,63           <134>
20 DATA 51,54,60,56,56,124,118,115         <000>
21 DATA 48,48,48,48,48,115,127,127         <170>
22 DATA 49,59,63,53,49,113,113,113         <238>
23 DATA 51,59,63,55,51,115,115,115         <113>
24 DATA 63,63,51,51,51,115,127,127         <248>
25 DATA 62,63,51,63,62,112,112,112         <173>
26 DATA 63,63,51,51,55,118,127,123         <238>
27 DATA 63,63,51,63,63,124,118,115         <030>
28 DATA 30,63,56,30,7,115,127,126          <048>
29 DATA 63,63,12,12,12,28,28,28            <078>
30 DATA 51,51,51,51,51,115,127,127         <149>
31 DATA 51,51,51,51,51,115,126,124         <101>
32 DATA 49,49,49,49,53,127,123,113         <072>
33 DATA 51,51,51,51,12,115,115,115         <236>
34 DATA 102,102,102,24,24,120,126,126      <246>
35 DATA 63,63,6,12,24,112,127,127          <224>
50 DATA 128,175,255,255,31,31,255,127      <149>
60 DATA 119,121,119,126,127,121,127,127    <246>
70 DATA 127,127,127,127,127,127,127,127    <184>
80 DATA 127,126,255,191,191,209,255,255    <094>
90 DATA 0,223,223,159,191,255,255,129      <223>
91 DATA 0,0,0,0,0,0,0,0                    <009>
100 DATA 128,128,128,0,129,255,255,255     <250>
109 DATA 0,0,0,0,0,0,0,0                   <027>
110 DATA 255,255,255,255,255,129,128,128   <239>
120 DATA 128,0,1,255,255,255,255,255       <030>
130 DATA 0,252,255,255,251,247,255,255     <020>
140 DATA 255,255,255,255,255,255,255,254   <220>
150 DATA 252,248,255,255,191,223,223,255   <206>
160 DATA 255,255,255,127,255,255,191,156   <171>
170 DATA 3,3,3,131,195,199,231,231         <092>
180 DATA 224,227,231,239,207,159,159,31    <031>
190 DATA 31,31,15,159,223,255,255,255      <023>
200 DATA 255,255,255,207,207,135,3,0       <154>
210 DATA 227,225,255,255,255,255,255,178   <184>
220 DATA 255,255,255,255,253,248,248,248   <073>
225 DATA 248,248,248,248,248,248,248,248   <139>
230 DATA 126,126,102,102,102,102,126,126   <124>
231 DATA 12,28,60,124,12,12,12,12          <052>
232 DATA 126,126,102,110,28,56,126,126     <166>
233 DATA 126,126,6,62,62,6,126,126         <198>
234 DATA 96,96,108,126,126,12,12,12        <050>
235 DATA 126,126,96,126,126,6,126,126      <148>
236 DATA 126,126,96,126,126,102,126,126    <004>
237 DATA 126,126,102,14,28,56,48,48        <107>
238 DATA 126,126,102,126,126,102,126,126   <046>
239 DATA 126,126,102,126,126,6,126,126     <081>
240 DATA 248,253,255,255,223,215,243,243   <198>
250 DATA 255,255,254,254,231,195,129,0     <182>
260 DATA 248,254,255,255,249,249,255,255   <031>
270 DATA 255,255,239,247,255,255,255,255   <198>
280 DATA 255,255,255,191,191,255,254,248   <132>
290 DATA 255,255,127,54,127,247,231,0      <121>
300 DATA 1,7,15,159,159,255,255,255        <089>
310 DATA 255,255,255,239,239,223,255,255   <234>
320 DATA 255,255,255,159,159,15,7,1        <084>
330 DATA 143,251,252,255,127,95,207,0      <167>
340 DATA 255,255,255,255,191,251,241,241   <191>
350 DATA 225,225,225,225,161,193,225,225   <032>
360 DATA 241,251,255,223,191,255,255,255   <213>
370 DATA 240,248,248,248,248,240,224,0     <199>
380 DATA 240,253,255,255,239,239,247,255   <179>
390 DATA 255,255,255,255,223,255,247,255   <154>
400 DATA 255,255,255,247,247,254,252,240   <170>
410 DATA 0,0,3,15,31,31,31,31              <169>
420 DATA 31,255,255,247,251,255,255,255    <118>
```

Listing 1. Lader zum Programm »Das Boot«

```
430 DATA 159,159,159,159,191,189,191,191        <119>
440 DATA 191,175,175,63,63,31,15,3              <048>
450 DATA 0,120,200,248,152,248,248,248          <087>
460 DATA 255,255,255,207,255,255,255,252        <025>
470 DATA 248,248,240,248,248,240,248,248        <113>
480 DATA 248,255,255,255,253,205,243,0          <080>
490 DATA 128,128,128,128,128,128,128,0          <200>
500 DATA 0,128,128,0,128,128,128,0              <018>
510 DATA 0,7,1,1,7,3,3,0                        <041>
520 DATA 3,3,3,3,3,3,3,3                        <184>
530 DATA 0,231,255,255,255,243,243,243          <111>
540 DATA 115,243,243,243,243,243,243,227        <044>
550 DATA 0,128,192,224,240,240,144,241          <180>
560 DATA 247,255,255,255,255,240,227,199        <070>
570 DATA 0,0,0,0,0,0,248,252                    <223>
580 DATA 254,255,255,191,31,255,255,255         <051>
590 DATA 0,0,0,0,0,0,15,31                      <214>
600 DATA 63,127,255,252,254,254,255,191         <013>
610 DATA 0,0,0,0,0,0,224,240                    <173>
611 DATA 0,0,0,0,0,0,0,0                        <021>
620 DATA 248,248,248,248,248,0,0,192            <192>
10020 DATA 0,0,3,63,117,63,0,0                  <227>
10022 DATA 0,196,200,254,87,254,0,0             <161>
10024 DATA 0,0,1,31,58,31,0,0                   <210>
10026 DATA 0,98,228,255,171,255,0,0             <079>
10028 DATA 0,0,0,0,128,0,0,0                    <195>
10030 DATA 0,0,0,15,29,15,0,0                   <087>
10032 DATA 0,49,242,255,85,255,0,0              <051>
10034 DATA 0,0,0,128,192,128,0,0                <098>
10036 DATA 0,0,0,7,14,7,0,0                     <194>
10038 DATA 0,24,121,255,170,255,0,0             <088>
10040 DATA 0,128,0,192,224,192,0,0              <194>
10042 DATA 0,0,0,3,7,3,0,0                      <212>
10044 DATA 0,12,60,255,85,255,0,0               <194>
10046 DATA 0,64,128,224,112,224,0,0             <184>
10048 DATA 0,0,0,1,3,1,0,0                      <118>
10050 DATA 0,6,30,255,170,255,0,0               <038>
10052 DATA 0,32,64,240,184,240,0,0              <143>
10054 DATA 0,0,0,0,1,0,0,0                      <074>
10056 DATA 0,3,15,255,213,255,0,0               <016>
10058 DATA 0,16,32,248,92,248,0,0               <200>
10100 DATA 0,3,63,117,63,0,0,0                  <026>
10102 DATA 196,200,254,87,254,0,0,0             <219>
10104 DATA 0,0,0,0,0,0,0,196                    <011>
10106 DATA 3,63,117,63,0,0,0,0                  <026>
10108 DATA 200,254,87,254,0,0,0,0               <073>
10110 DATA 0,0,0,0,0,0,0,3                      <128>
10112 DATA 0,0,0,0,0,0,196,200                  <095>
10114 DATA 63,117,63,0,0,0,0,0                  <155>
10116 DATA 254,87,254,0,0,0,0,0                 <152>
10118 DATA 0,0,0,0,0,0,3,63                     <219>
10120 DATA 0,0,0,0,0,196,200,254                <158>
10126 DATA 117,63,0,0,0,0,0,0                   <148>
10128 DATA 87,254,0,0,0,0,0,0                   <206>
10130 DATA 0,0,0,0,0,3,63,117                   <176>
10132 DATA 0,0,0,0,196,200,254,87               <166>
10134 DATA 63,0,0,0,0,0,0,0                     <024>
10136 DATA 254,0,0,0,0,0,0,0                    <230>
10138 DATA 0,0,0,0,3,63,117,63                  <058>
10140 DATA 0,0,0,196,200,254,87,254             <208>
10142 DATA 0,0,0,3,63,117,63,0                  <189>
10144 DATA 0,0,196,200,254,87,254,0             <115>
20000 DATA 0,0,0,45,111,255,255,255             <240>
20002 DATA 0,8,24,59,127,255,255,255            <123>
20004 DATA 0,0,0,144,248,252,252,254            <210>
20006 DATA 254,254,252,248,252,254,254,252      <015>
20008 DATA 252,248,240,224,192,252,252,254      <118>
20010 DATA 252,254,255,255,255,255,255,255      <182>
20012 DATA 31,127,127,255,255,255,255,255       <155>
20014 DATA 63,127,63,63,127,15,31,63            <057>
20016 DATA 15,31,15,7,1,3,7,31                  <221>
20018 DATA 63,127,63,63,63,127,45,0             <100>
20020 DATA 255,254,252,254,255,254,126,34       <185>
20022 DATA 17,63,127,63,127,255,127,63          <153>
20024 DATA 255,255,255,255,254,188,24,8         <168>
20026 DATA 195,36,24,255,90,255,0,0             <099>
20028 DATA 255,255,255,255,255,255,255,255      <042>
21002 DATA 0,0,4,61,113,55,0,0                  <154>
21004 DATA 2,68,128,222,71,246,0,0              <110>
21006 DATA 192,97,0,13,65,237,192,0             <048>
21008 DATA 48,3,7,174,36,128,48,24              <037>
21010 DATA 48,97,192,1,36,116,48,0              <093>
21012 DATA 130,198,128,33,18,0,6,6              <113>
21013 DATA 0,0,0,0,0,0,0,0                      <103>
21014 DATA 31,43,85,251,173,250,172,248         <209>
21020 REM VERG. U-BOOT                          <230>
21030 DATA 0,136,144,252,174,252,0,0            <203>
21040 DATA 0,1,7,127,234,127,0,0                <232>
21100 DATA 0,0,24,60,24,0,0,0                   <071>
21110 DATA 132,68,36,18,11,7,207,63             <120>
21120 DATA 65,66,132,136,208,224,243,252        <067>
21130 DATA 15,31,39,203,18,36,72,136            <166>
21140 DATA 240,248,228,211,72,68,34,33          <032>
21150 DATA 15,30,60,124,120,248,240,240         <189>
21160 DATA 240,240,248,120,124,60,30,15         <190>
21170 DATA 16,16,56,254,56,16,16,0              <172>
21180 DATA 0,0,0,0,4,46,127,255                 <138>
21190 DATA 1,1,3,71,79,239,255,255              <172>
21200 DATA 128,128,192,226,224,247,255,255      <251>
21210 DATA 129,131,195,225,231,239,239,255      <138>
21220 DATA 3,15,47,119,251,253,253,254          <029>
21230 DATA 0,60,126,255,255,255,255,127         <185>
21240 DATA 0,60,126,127,255,255,255,255         <195>
21250 DATA 0,60,126,255,255,255,255,254         <127>
21260 DATA 127,255,255,255,255,126,60,0         <130>
21270 DATA 255,255,255,255,254,124,56,0         <184>
21280 DATA 254,255,255,255,255,126,60,0         <159>
```


Listing 1. Lader zum Programm »Das Boot« (Schluß)

```
0 DIM X$(17),Y$(20),Z$(23),SN(176),SD(133)
  ,SA(66),SQ(92)                                <203>
10 QW=36878:ER=36877:TY=36875:UI=36879:LE=
  1:XY=4102:YX=4103:A$=CHR$(147):ZX=36876       <063>
11 KL=36874:HI=750:P1=500                       <081>
200 PRINT A$:POKE UI,8:POKE 36869,205:GOSU
  B 229                                         <138>
210 GOSUB 5000                                  <198>
220 GOSUB 52000:GOSUB 52100:GOSUB 53000:GO
  SUB 52600:GOTO 315                            <119>
229 X$(0)="{CLR,BLACK,RVOFF}TTTTTT SC{SPAC
  E,RVOFF,4SPACE}HI{6SPACE}"                    <173>
230 X$(1)="{3DOWN,10SPACE,RVSON}123{RVOFF,
  4SPACE,RVSON}%&{RVOFF,3SPACE}"                <023>
231 X$(2)="{10SPACE,RVSON}456{RVOFF,4SPACE
  ,RVSON}'({RVOFF,3SPACE}"                      <052>
232 X$(3)="{2DOWN,2SPACE,RVSON}123{RVOFF,4
  SPACE,RVSON}-0.{RVOFF,10SPACE}"               <133>
233 X$(4)="{2SPACE,RVSON}456{RVOFF,3SPACE,
  RVSON}-YYY.,-.{RVOFF,SPACE,RVSON},{RVO
  FF,4SPACE}"                                   <133>
234 X$(5)="{7SPACE,RVSON}-YYYYYYYY/Y.{RVOF
  F,3SPACE}"                                    <181>
235 X$(6)="{5SPACE,RVSON}-0YYYYYYYYYYYY.,,
  {RVOFF}"                                      <125>
236 X$(7)="{SPACE,RVSON},-.-YYYYYYYYYYYYY
  YYY{RVOFF}"                                   <024>
237 X$(8)="{RVSON}-WWWWWYYWWWWWWWYYYWWWW{R
  VOFF}"                                        <028>
238 X$(9)="@@@@@@{RVSON}RNX{RVOFF}@@@@@@{R
  VSON}SYO{RVOFF}@@@ "                          <209>
239 X$(10)="{RVSON}KKMX{RVOFF}@@{RVSON}SO{
  RVOFF}@@@{RVSON}X{RVOFF}@@@{RVSON}RYO{
  RVOFF}@@{RVSON}KK{RVOFF}"                     <165>
240 X$(11)="{RVSON}YYN{RVOFF}@@@{RVSON}RN{
  RVOFF}@@@{RVSON}VP{RVOFF}@@{RVSON}SYN{
  RVOFF}@@{RVSON}RY{RVOFF}"                     <062>
241 X$(13)="{RVSON}YYN{RVOFF}@@@{RVSON}TUX
  {RVOFF}@@{RVSON}RN{RVOFF}@@{RVSON}TWU{
  RVOFF}@@{RVSON}RY{RVOFF,RVSON}YYO{RVOF
  F}@@@@@@@@{RVSON}SO{RVOFF}@@@@@@@{RVSO
  N}RY{RVOFF}"                                  <188>
242 X$(12)="{RVSON}YYO{RVOFF}@@{RVSON}XSO{
  RVOFF}@@@{RVSON}RN{RVOFF}@@{RVSON}RYO{
  RVOFF}@@{RVSON}SY{RVOFF}"                     <216>
243 X$(14)="{RVSON}YYPKLKLM{RVOFF}@@{RVSON
  }VQO{RVOFF}@@{RVSON}X{RVOFF}@@@@{RVSON
  }RY{RVOFF}"                                   <147>
```

Listing 2. Hauptprogramm zum Superspiel »Das Boot«

```
244 X$(15)=" {RVSON}YYYYYYYN!!RYN{RVOFF}@@@
    @@@@{RVSON}SY{RVOFF}"                     <019>
245 X$(16)=" {RVSON}YYYYYYYYKLYYPLKLKLKKQ"    <119>
246 Y$(0)=" {CLR,CYAN,RVOFF}TTTTTT SC{SPACE
    ,RVOFF,4SPACE}HI{6SPACE}"                 <254>
247 Y$(1)=" {SPACE,RVSON}${RVOFF,20SPACE}"   <084>
248 Y$(2)=" {5SPACE,RVSON}${RVOFF,7SPACE,RV
    SON}+{RVOFF,6SPACE,RVSON}${RVOFF,SPACE
    }"                                        <087>
249 Y$(3)=" {DOWN,3SPACE,RVSON}){RVOFF,5SPA
    CE,RVSON}${RVOFF,6SPACE,RVSON}${RVOFF,
    5SPACE}"                                  <081>
250 Y$(4)=" {3SPACE,RVSON}*${RVOFF,12SPACE,
    RVSON}+{RVOFF,4SPACE}"                    <218>
251 Y$(5)=" {DOWN,SPACE,RVSON}+{RVOFF,18SPA
    CE,RVSON}+-{RVOFF}"                       <210>
252 Y$(6)=" {5SPACE,RVSON}${RVOFF,SPACE,RVS
    ON}+{RVOFF,10SPACE,RVSON}${RVOFF,SPACE
    ,RVSON}-Y{RVOFF}"                         <115>
253 Y$(7)=" {10SPACE,RVSON}-/.{RVOFF,4SPACE
    ,RVSON}-,0YY{RVOFF}"                      <119>
254 Y$(8)=" {RVSON}.{RVOFF,SPACE,RVSON}${RV
    OFF,2SPACE,RVSON}-.{RVOFF,2SPACE,RVSON
    }-YYY.,,0YYYYY{RVOFF}"                    <117>
255 Y$(9)=" {RVSON}Y.,,-YY.-YYYYYYYYYYYYY{R
    VOFF}"                                    <242>
256 Y$(10)=" {RVSON}YWWYYWWYYWWWWWWWWWWWWY{
    RVOFF}"                                   <077>
257 Y$(11)=" {RVSON}N{RVOFF}@@{RVSON}SN{RVO
    FF}@@{RVSON}RO{RVOFF}@@@@@@@@@@@@{RVSO
    N}R{RVOFF}"                               <032>
258 Y$(12)=" {RVSON}N{RVOFF}@@{RVSON}TU{RVO
    FF}@@{RVSON}SO{RVOFF}@@{RVSON}VML{RVOF
    F}@@{RVSON}KLL{RVOFF}@@{RVSON}S{RVOFF}
    "                                         <062>
259 Y$(13)=" {RVSON}O{RVOFF}@@@@@{RVSON}XRN
    {RVOFF}@@{RVSON}SYO!!SYN{RVOFF}@@{RVSO
    N}S{RVOFF}"                               <114>
260 Y$(14)=" {RVSON}N{RVOFF}@@{RVSON}X{RVOF
    F}@@@{RVSON}RO{RVOFF}@@{RVSON}SYYYYYYO
    {RVOFF}@@{RVSON}R{RVOFF}"                 <008>
261 Y$(15)=" {RVSON}O{RVOFF}@@{RVSON}VM{RVO
    FF}@@{RVSON}TU{RVOFF}@@{RVSON}RYYWWWWU
    {RVOFF}@@{RVSON}S{RVOFF}"                 <163>
262 Y$(16)=" {RVSON}O{RVOFF}@@{RVSON}SN{RVO
    FF}@@@@@@{RVSON}SYO{RVOFF}@@@@@@@{RVSO
    N}R{RVOFF}"                               <201>
263 Y$(17)=" {RVSON}N{RVOFF}@@{RVSON}RO{RVO
    FF}@@@@@@{RVSON}RYN{RVOFF}@@@{RVSON}VL
    KLY{RVOFF}"                               <004>
264 Y$(18)=" {RVSON}O{RVOFF}@@{RVSON}SYLKKM
    {RVOFF}@@{RVSON}RYO{RVOFF}@@@{RVSON}TW
    WWW{RVOFF}"                               <136>
265 Y$(19)=" {RVSON}N{RVOFF}@@{RVSON}SYYYYO
    !!SYO{RVOFF}@@@@@@@ "                     <131>
266 Y$(20)=" {RVSON}YKLYYYYYYKLYYYLKLLKLL{R
    VOFF}"                                    <118>
267 Z$(0)=" {CLR,YELLOW,RVOFF}TTTTTT SC{SPA
    CE,RVOFF,4SPACE}HI{6SPACE}"               <021>
268 Z$(1)=" {RVSON}WWWWWWWYYYYYYYYYYYYYYY{R
    VOFF}"                                    <237>
269 Z$(2)="@@@@@@{RVSON}TWWWWWWYYWWWWWYY{R
    VOFF}"                                    <028>
270 Z$(3)="@@{RVSON}QM{RVOFF}@@@@@@@@@{RVS
    ON}TU{RVOFF}@@@@@{RVSON}RY{RVOFF}"        <156>
271 Z$(4)=" {RVSON}KLYN{RVOFF}@@@@@{RVSON}V
    P{RVOFF}@@@@@@@@@{RVSON}SY{RVOFF}"        <047>
272 Z$(5)=" {RVSON}YWWU{RVOFF}@@{RVSON}VKLY
    YKLLKKLM{RVOFF}@@{RVSON}SY{RVOFF}"        <052>
273 Z$(6)=" {RVSON}N{RVOFF}@@@@@{RVSON}RYYY
    YYYYYYYO{RVOFF}@@{RVSON}TW{RVOFF}"        <104>
274 Z$(7)=" {RVSON}O{RVOFF}@@{RVSON}KLKYYYY
    YYYYYYYU{RVOFF}@@@ "                      <077>
275 Z$(8)=" {RVSON}N{RVOFF}@@{RVSON}TWWWWWY
    YWWWYYN!!VLK{RVOFF}"                      <065>
276 Z$(9)=" {RVSON}O{RVOFF}@@@@@@@@{RVSON}T
    U{RVOFF}@@@{RVSON}RYO{RVOFF}@@{RVSON}S
    YY{RVOFF}"                                <031>
277 Z$(10)=" {RVSON}N{RVOFF}@@@@{RVSON}QM{R
    VOFF}@@@@@@@{RVSON}SYN{RVOFF}@@{RVSON}
    RYY{RVOFF}"                               <237>
278 Z$(11)=" {RVSON}O{RVOFF}@@{RVSON}VLYYKL
    KLM{RVOFF}@@{RVSON}RYO{RVOFF}@@{RVSON}
    SYY{RVOFF}"                               <031>
279 Z$(12)=" {RVSON}N{RVOFF}@@{RVSON}SYYYYY
    YYN{RVOFF}@@{RVSON}SYN{RVOFF}@@{RVSON}
    RYY{RVOFF}"                               <141>
280 Z$(13)=" {RVSON}N{RVOFF}@@{RVSON}SYWWYW
    WWU{RVOFF}@@{RVSON}SYN{RVOFF}@@{RVSON}
    SYY{RVOFF}"                               <010>
281 Z$(14)=" {RVSON}O{RVOFF}@@{RVSON}RN!!S!
    !{RVOFF}@@@@{RVSON}RYO{RVOFF}@@{RVSON}
    SYY{RVOFF}"                               <068>
282 Z$(15)=" {RVSON}N{RVOFF}@@{RVSON}SO{RVO
    FF}@@{RVSON}RKLLKLKQYN{RVOFF}@@{RVSON}
    RYY{RVOFF}"                               <103>
283 Z$(16)=" {RVSON}O{RVOFF}@@{RVSON}TU{RVO
    FF}@@{RVSON}TWWWWWWYYO{RVOFF}@@{RVSON}
    SYY{RVOFF}"                               <063>
284 Z$(17)=" {RVSON}N{RVOFF}@@@@@@@@@@@@@{R
    VSON}TWU{RVOFF}@@{RVSON}RYY{RVOFF}"       <052>
285 Z$(18)=" {RVSON}O!!VLKMVM{RVOFF}@@@@@@@
    @@@{RVSON}TYY{RVOFF}"                     <174>
286 Z$(19)=" {RVSON}YKLYYYYYN{RVOFF}@@{RVSO
    N}VKLLKLKM{RVOFF}@{RVSON}SY{RVOFF}"       <062>
287 Z$(20)=" {RVSON}YYYYYYYYN{RVOFF}@@{RVSO
    N}SYYYYYYYLYY{RVOFF}"                     <024>
288 Z$(21)=" {RVSON}YYYYYYYYO!!RYYYYYYYYY{
    RVOFF}"                                   <042>
289 Z$(22)=" {RVSON}YYYYYYYYYKLYYYYYYYYYY{R
    VOFF}":RETURN                             <006>
300 FOR W=0 TO 22:PRINT Z$(W);:NEXT:POKE 4
    601,153:END                               <173>
315 IF LE=1 THEN PRINT A$:POKE UI,8:GOTO 5
    2300                                      <197>
316 IF LE=2 THEN PRINT A$:POKE UI,8:GOTO 6
    0000                                      <142>
317 IF LE=3 THEN PRINT A$:POKE UI,8:GOTO 5
    2400                                      <103>
318 IF LE=4 THEN PRINT A$:POKE UI,8:GOTO 6
    1000                                      <033>
319 IF LE=5 THEN PRINT A$:POKE UI,8:GOTO 5
    2500                                      <010>
320 IF LE=6 THEN PRINT A$:POKE UI,8:GOTO 6
    2700                                      <179>
730 POKE 37154,127:IF PEEK(37152)=119 THEN
    GOTO 1100                                 <064>
740 POKE 37154,255:J=PEEK(37151):IF(J AND
    4)=.THEN GOTO 800                         <130>
750 IF(J AND 8)=.THEN GOTO 900                <057>
760 IF(J AND 16)=.THEN GOTO 1000              <000>
792 GOTO 730                                  <054>
800 IF PEEK(A-22)=161 THEN GOSUB 2500:GOSU
    B 3000:GOTO 730                           <094>
820 IF PEEK(A-22)OR PEEK(B-22)<>0 THEN GOT
    O 1500                                    <121>
840 D=123                                     <025>
850 POKE A,118:POKE B,119:POKE B-22,120:PO
    KE A,121:POKE B,122                       <190>
855 FOR T=1 TO 3:POKE A-22,D:POKE B-22,D+1
    :POKE A,D+2:POKE B,D+3:D=D+4:NEXT:POKE
    A,0:POKE B,0:A=A-22                       <002>
857 B=B-22                                    <059>
860 POKE A,135:POKE B,136:POKE A,137:POKE
    B,138:POKE A,98:POKE B,99:GOSUB 4000:G
    OTO 1700                                  <208>
900 IF PEEK(A+22)=161 THEN GOSUB 2600:GOSU
    B 3000:GOTO 730                           <010>
910 IF PEEK(A+22)OR PEEK(B+22)<>0 THEN GOT
    O 1500                                    <083>
930 F=131                                     <027>
935 POKE A,137:POKE B,138:POKE A,135:POKE
    B,136                                     <185>
940 FOR T=1 TO 3:POKE A,F:POKE B,F+1:POKE
    A+22,F+2:POKE B+22,F+3:F=F-4:NEXT         <097>
945 POKE A,0:POKE B,0:A=A+22:B=B+22:POKE A
    ,118:POKE B,119:POKE A,98:POKE B,99:GO
    SUB 4000:GOTO 1800                        <052>
1000 IF PEEK(A-1)=161 THEN GOSUB 2700:GOSU
     B 3000:GOTO 730                          <042>
1020 IF PEEK(A-1)<>0 THEN GOTO 1500           <106>
1030 E=117                                    <190>
1035 POKE B,162:POKE A,163                    <133>
1040 FOR T=1 TO 6:POKE B,E:POKE A,E-1:POKE
     A-1,E-2:E=E-3:NEXT                       <095>
```

Listing 2. Hauptprogramm zum Superspiel »Das Boot« (Fortsetzung)

```
1050 POKE B,0:A=A-1:B=B-1:POKE A,98:POKE B
     ,99                                    <236>
1060 GOSUB 4000:GOTO 1900                   <136>
1100 IF PEEK(B+1)=32 THEN LE=LE+1:GOTO 315 <178>
1110 IF PEEK(B+1)<>0 THEN GOTO 1500         <166>
1119 C=100                                  <046>
1120 FOR T=1 TO 6:POKE A,C:POKE B,C+1:POKE
      B+1,C+2:C=C+3:NEXT:POKE A,0           <135>
1125 A=A+1:B=B+1:POKE A,163:POKE B,162      <092>
1130 POKE A,98:POKE B,99:GOSUB 4000:GOTO 2
     000                                    <044>
1500 POKE A,154:POKE B,155:POKE QW,10:POKE
      ER,151:FOR I=1 TO 400:NEXT:POKE QW,0 <210>
1501 POKE A,156:POKE B,157:POKE QW,5        <095>
1502 FOR T=1 TO 400:NEXT                    <176>
1503 POKE A,158:POKE B,159:POKE QW,15:FOR
     T=0 TO 400:NEXT:POKE QW,0:POKE A,0:PO
     KE B,0:POKE ER,0                       <015>
1510 IF LE=1 THEN A=4404:B=4405:POKE A,98:
     POKE B,99                              <018>
1520 IF LE=3 THEN A=4559:B=4560:POKE A,98:
     POKE B,99                              <121>
1530 IF LE=5 THEN A=4162:B=4163:POKE A,98:
     POKE B,99                              <051>
1532 P1=P1-100:PRINT"{HOME,9RIGHT,PURPLE}"
     ;P1                                    <212>
1544 XY=XY-2:YX=YX-2:POKE XY,0:POKE YX,0
                                            <197>
1549 IF XY=4096 THEN GOSUB 54000            <245>
1550 GOTO 730                               <050>
1700 POKE 37154,127:IF PEEK(37152)=119 THE
     N GOTO 1100                            <018>
1710 POKE 37154,255:J=PEEK(37151)           <109>
1720 IF(J AND 8)=.THEN GOTO 900             <011>
1730 IF(J AND 16)=.THEN GOTO 1000           <210>
1740 GOTO 800                               <200>
1800 POKE 37154,127:IF PEEK(37152)=119 THE
     N GOTO 1100                            <120>
1810 POKE 37154,255:J=PEEK(37151):IF(J AND
      4)=.THEN GOTO 800                     <186>
1820 IF(J AND 16)=.THEN GOTO 1000           <046>
1830 GOTO 900                               <044>
1900 POKE 37154,127:IF PEEK(37152)=119 THE
     N GOTO 1100                            <220>
1910 POKE 37154,255:J=PEEK(37151):IF(J AND
      4)=.THEN GOTO 800                     <030>
1920 IF(J AND 8)=.THEN GOTO 900             <213>
1930 GOTO 1000                              <092>
2000 POKE 37154,127                         <228>
2010 POKE 37154,255:J=PEEK(37151):IF(J AND
      4)=.THEN GOTO 800                     <130>
2020 IF(J AND 8)=.THEN GOTO 900             <057>
2030 IF(J AND 16)=.THEN 1000                <043>
2040 GOTO 1100                              <218>
2500 A=A-22:B=B-22:POKE A,98:POKE B,99:POK
     E A+22,0:POKE B+22,0:RETURN            <156>
2600 A=A+22:B=B+22:POKE A,98:POKE B,99:POK
     E A-22,0:POKE B-22,0:RETURN            <082>
2700 A=A-2:B=B-2:POKE A,98:POKE B,99:POKE
     A+2,0:POKE B+2,0:RETURN       0        <109>
3000 FOR T=0 TO 19:POKE QW,15:POKE TY,220:
     POKE TY,0                              <089>
3010 POKE TY,200:POKE TY,0:POKE QW,0:P1=P1
     +10:PRINT"{HOME,9RIGHT,PURPLE}";P1:NE
     XT                                     <088>
3020 POKE A,98:POKE B,99:RETURN             <111>
4000 P1=P1-2:PRINT"{HOME,9RIGHT,PURPLE}";P
     1:IF P1<5 THEN GOTO 54000              <026>
4010 RETURN                                 <002>
5000 FOR I=0 TO 175:READ SN(I):NEXT I       <082>
5100 FOR I=0 TO 132:READ SD(I):NEXT I       <107>
5200 FOR I=0 TO 65:READ SA(I):NEXT I        <001>
5300 FOR I=0 TO 91:READ SQ(I):NEXT I:RETUR
     N                                      <156>
6000 POKE QW,15:POKE ER,200:FOR I=1 TO 20:
     NEXT:POKE ER,0                         <136>
6005 POKE ER,180:FOR I=1 TO 20:NEXT:POKE E
     R,0:POKE QW,0:RETURN                   <075>
7000 FOR J=0 TO 3000:NEXT                   <034>
7003 PRINT"{CLR,11DOWN,YELLOW,20SPACE}":FO
     R T=0 TO 1200:NEXT:GOTO 52000          <244>
8000 DATA 0,18,15,0,13,1,0,19,15,6,20,0,16
     ,18,5,19,5,14,20,19,0,0                <121>
8010 DATA 0,0,0,0,0,0,1,0,7,1,13,5,0,2,25,
     0,0,0,0,0,0,0                          <244>
8020 DATA 0,0,0,0,18,15,7,5,18,0,4,5,19,9,
     4,5,18,9,0,0,0,0                       <248>
8030 DATA 0,0,0,0,0,0,0,0,1,14,4,0,0,0,0
     ,0,0,0,0,0,0                           <253>
8040 DATA 0,0,0,0,13,1,18,20,9,14,0,7,18,2
     6,9,2,5,11,0,0,0,0                     <045>
8050 DATA 0,13,21,19,9,3,0,20,18,1,14,19,1
     2,1,20,5,4,0,2,25,0,0                  <058>
8060 DATA 0,0,0,14,15,18,2,5,18,20,0,7,18,
     26,9,2,5,11,0,0,0,0                    <050>
8070 DATA 0,0,0,16,18,5,19,19,0,6,9,18,5,0
     ,2,21,20,20,15,14,0,0                  <229>
9000 DATA 0,0,0,0,0,0,0,0,85,87,89,91,93
     ,95,0,0,0,0,0,0,0                      <026>
9010 DATA 0,0,0,0,0,0,0,0,86,88,90,92,94
     ,97,0,0,0,0,0,0,0                      <163>
9020 DATA 0,0,0,0,0,0,27,31,37,41,45,59,63
     ,67,71,75,79,0,0,0,0,0                 <141>
9030 DATA 0,0,0,0,0,0,28,33,38,42,46,60,64
     ,68,72,76,80,83,0,0,0,0                <135>
9040 DATA 0,0,0,0,0,0,29,35,39,43,47,61,65
     ,69,73,77,81,0,0,0,0,0                 <030>
9050 DATA 0,0,0,0,0,0,30,36,40,44,58,62,66
     ,70,74,78,82,84,0,0,0,0                <033>
10000 DATA 0,0,0,0,0,0,12,5,22,5,12,0,0,0,
      9,0,0,0,0,0,0,0                       <040>
10010 DATA 0,0,0,0,0,0,12,5,22,5,12,0,0,9,
      9,0,0,0,0,0,0,0                       <059>
10020 DATA 0,0,0,0,0,0,12,5,22,5,12,0,9,9,
      9,0,0,0,0,0,0,0,0                     <159>
11000 DATA 151,1000,147,250,135,250,147,25
      0,151,250,175,250,187,1000,-3,151,10
      00,147,250,135                        <072>
11010 DATA 250,147,250,151,250,175,250,187
      ,500,183,250,175,250,183,250,187,250
      ,201,250                              <083>
11020 DATA 209,1000,-3,151,1000,147,250,13
      5,250,147,250,151,250,175,250,187,50
      0,183,250                             <244>
11021 DATA 175,250                          <204>
11030 DATA 183,250,187,250,201,250,209,250
      ,207,250,201,250,207,250,209,250,219
      ,250,225,250                          <075>
11040 DATA 223,250,219,250,223,250,225,250
      ,231,250,235,0                        <146>
50000 A=4404:B=4405:POKE A,98:POKE B,99:GO
      TO 730                                <083>
52000 A=4338:PRINT A$:FOR F=38130 TO 38152
      :POKE F,7:NEXT                        <236>
52001 FOR T=0 TO 21:POKE A+T,SN(K):K=K+1:I
      F K=153 THEN FOR U=0 TO 3000:NEXT:RE
      TURN                                  <209>
52002 IF SN(K)=0 THEN NEXT:GOTO 7000        <221>
52003 GOSUB 6000:NEXT                       <085>
52021 POKE 38191,3:POKE 38192,2:POKE 38194
      ,7:POKE 38195,5                       <004>
52100 P1=500:POKE 36865,155:D=4294:PRINT A
      $:FOR F=38086 TO 38218:POKE F,7:NEXT <036>
52110 FOR Y=0 TO 131:POKE D+Y,SD(Y):NEXT:P
      OKE 36865,38:POKE QW,15:FOR F=230 TO
       140 STEP-1                           <015>
52120 POKE QW-2,F:POKE QW-3,F+1:NEXT:POKE
      QW-4,140:FOR F=1 TO 500:NEXT:FOR F=1
      5 TO 0 STEP-.007:POKE QW,F:NEXT       <124>
52130 POKE QW-4,0:POKE QW-3,0:POKE QW-2,0:
      RETURN                                <124>
52300 H=4339:FOR F=38130 TO 38152:POKE F,7
      :NEXT                                 <157>
52310 FOR B=0 TO 21:POKE H+B,SA(B):IF SA(B
      )=0 THEN 52330                        <107>
52320 GOSUB 6000                            <245>
52330 NEXT                                  <013>
52399 GOSUB 58000:GOTO 52700                <011>
52400 H=4340:FOR F=38130 TO 38152:POKE F,7
      :NEXT                                 <190>
52410 FOR B=22 TO 43:POKE H+B-22,SA(B):IF
      SA(B)=0 THEN 52430                    <001>
```

Listing 2. Hauptprogramm zum Superspiel »Das Boot« (Fortsetzung)

```
52420 GOSUB 6000                                  <089>
52430 NEXT:GOSUB 58000:GOTO 52800                 <153>
52500 H=4338:FOR F=38130 TO 38152:POKE F,7
      :NEXT                                       <231>
52510 FOR B=44 TO 60:POKE H+B-43,SA(B):IF
      SA(B)=0 THEN 52530                          <056>
52520 GOSUB 6000                                  <191>
52530 NEXT:GOSUB 58000:GOTO 52900                 <063>
52600 H=4338:FOR F=38284 TO 38306:POKE F,7
      :NEXT                                       <147>
52610 FOR B=154 TO 175:POKE H+B,SN(B):IF S
      N(B)=0 THEN 52630                           <121>
52620 GOSUB 6000                                  <035>
52630 NEXT                                        <059>
52640 IF PEEK(37154-3)AND 32 THEN 52640           <239>
52650 RETURN                                      <127>
52700 AA=37888:CZ=4096:A=4404:B=4405:POKE
      36865,155                                   <237>
52710 FOR Y=0 TO 16:PRINT X$(Y);:NEXT:POKE
      4601,153:PRINT"{HOME,10RIGHT}500"           <082>
52712 FOR YU=37888 TO 37888+505:POKE YU,0:
      NEXT                                        <200>
52715 POKE A,98:POKE B,99                         <041>
52720 POKE AA+98,6:POKE AA+99,6:POKE AA+10
      0,6:POKE AA+120,6:POKE AA+121,6:POKE
      AA+122,6                                    <114>
52721 POKE AA+105,7:POKE AA+106,7:POKE AA+
      127,7:POKE AA+128,7:POKE AA+178,6:PO
      KE AA+179,6                                 <048>
52722 POKE AA+180,6:POKE AA+200,6:POKE AA+
      201,6:POKE AA+202,6                         <180>
52723 POKE 38221,7:POKE 38204,2:POKE 38229
      ,6:POKE 38267,4:POKE 38292,7:POKE 38
      343,2                                       <132>
52724 POKE 38358,4:POKE 38359,2:POKE 36865
      ,38:POKE UI,186:PRINT"{HOME,9RIGHT,P
      URPLE}";P1                                  <192>
52725 PRINT"{HOME,16RIGHT}";HI                    <175>
52799 GOTO 730                                    <244>
52800 XY=4102:YX=4103:QA=4096:POKE 36865,1
      55                                          <139>
52805 FOR T=0 TO 20:PRINT Y$(T);:NEXT:POKE
      4601,139                                    <104>
52806 FOR YU=37888 TO 37888+505:POKE YU,0:
      NEXT                                        <040>
52810 POKE 4559,98:POKE 4560,99:A=4559            <041>
52815 B=4560                                      <024>
52816 FOR U=38024 TO 38394:POKE U,5:NEXT          <195>
52820 POKE 37911,7:POKE 37937,7:POKE 37945
0     ,7:POKE 37952,7:POKE 37979,7                <037>
52821 POKE 37985,7:POKE 37992,7:POKE 38001
      ,7:POKE 38002,7:POKE 38015,7:POKE 38
      043,7                                       <160>
52822 POKE 38062,7:POKE 38082,7:POKE 38069
      ,7:POKE 38071,7:POKE 38110,7                <005>
52823 POKE 38243,3:POKE 38224,0                   <187>
52824 POKE 38232,7:POKE 38233,2:POKE 38359
      ,0:POKE 38360,1                             <009>
52897 PRINT"{HOME,9RIGHT,PURPLE}";P1:POKE
      36865,38:POKE UI,106                        <064>
52898 PRINT"{HOME,16RIGHT}";HI                    <094>
52899 GOTO 730                                    <088>
52900 XY=4102:YX=4103:QA=4096:POKE 36865,1
      55                                          <239>
52910 FOR W=0 TO 22:PRINT Z$(W);:NEXT:POKE
      4601,153                                    <099>
52912 FOR YU=37888 TO 37888+505:POKE YU,0:
      NEXT                                        <146>
52915 A=4162:B=4163:POKE A,98:POKE B,99           <200>
52920 FOR SD=37888 TO 38393:POKE SD,3:NEXT        <024>
52921 POKE 38081,5:POKE 38082,2:POKE 38201
      ,7:POKE 38202,5:POKE 38204,1:POKE 38
      205,4                                       <155>
52922 POKE 38286,4:POKE 38285,6:POKE 38359
      ,5:POKE 38360,1                             <102>
52998 PRINT"{HOME,9RIGHT,PURPLE}";P1:POKE
      36865,38:POKE UI,10                         <068>
52999 PRINT"{HOME,16RIGHT}";HI:GOTO 730           <081>
53000 D=0                                         <080>
53001 FOR P=3 TO 15 STEP.5:POKE QW,P              <203>
53009 IF SQ(D)=-3 THEN D=D+1:V=D:GOSUB 531
      10:GOTO 53001                               <076>
53010 POKE TY,SQ(D):POKE ZX,SQ(D):D=D+1           <179>
53012 FOR T=0 TO SQ(D):NEXT:D=D+1:FOR Q=0
      TO 300:NEXT                                 <153>
53097 IF SQ(D)=0 THEN GOSUB 53120:RETURN          <167>
53100 NEXT:GOTO 53009                             <170>
53110 FOR E=6 TO 0 STEP-.008:POKE QW,E:NEX
      T:POKE QW,0:POKE TY,0:POKE ZX,0:RETU
      RN                                          <024>
53120 FOR E=14 TO 0 STEP-.008:POKE QW,E:NE
      XT:POKE QW,0:POKE TY,0:POKE ZX,0:RET
      URN                                         <132>
54000 PRINT A$:POKE UI,8                          <158>
54010 PRINT"{10DOWN,7RIGHT,YELLOW}THE END{
      7RIGHT}"                                    <251>
54040 FOR Y=0 TO 1000:NEXT:LE=1:XY=4102:YX
      =4103:GOSUB 54100:GOSUB 52100:GOSUB
      52600:GOTO 315                              <153>
54100 IF P1>HI THEN HI=P1:PRINT"{CLR,DOWN,
      YELLOW}YOU HAVE BROKEN THE{DOWN,3SPA
      CE}HIGHSCORE":GOSUB 56000:GOSUB 5700
      0                                           <249>
54500 IF P1<HI THEN PRINT"{CLR,DOWN,YELLOW
      }YOU HAVE NOT REACHED{DOWN,2SPACE}TH
      E HIGHSCORE ":GOSUB 56000:GOTO 57000        <224>
54501 RETURN                                      <200>
56000 POKE QW,15:FOR L=1 TO 20:FOR M=220-L
       TO 210-L STEP-4:POKE ZX,M:NEXT M:FO
      R M=160-L TO 170-L STEP 4                   <008>
56001 POKE ZX,M:NEXT M:NEXT L:POKE QW,0:PO
      KE ZX,0                                     <048>
56002 FOR T=1 TO 3000:NEXT:XY=4102:YX=4103
      :RETURN                                     <075>
57000 PRINT"{DOWN,YELLOW}SCORE{4SPACE}";P1
      :GOSUB 56000                                <205>
57010 PRINT"{DOWN,YELLOW}THE HIGHSCORE ";H
      I:GOSUB 56000:RETURN                        <169>
58000 RA=151:RF=1                                 <125>
58001 A=4404:B=4405:PRINT"{12DOWN}":FOR X=
      1 TO 22:POKE 646,RND(1)*6+2:PRINT"{R
      VSON,SPACE}";:NEXT                          <033>
58010 FOR AS=4404 TO 4423:C=100                   <078>
58020 FOR T=1 TO 6:POKE A,C:POKE B,C+1:POK
      E B+1,C+2:C=C+3:GOSUB 58050:NEXT:POK
      E A,0:A=A+1:B=B+1:POKE A,163                <043>
58030 POKE B,162:GOSUB 58050:POKE A,98:POK
      E B,99:GOSUB 58050                          <027>
58040 POKE QW,RF:POKE TY,RA:RF=RF+.7:RA=RA
      +1:NEXT:POKE QW,0:POKE TY,0:RETURN          <089>
58050 FOR P=1 TO 25:NEXT:RETURN                   <014>
60000 GOSUB 63000                                 <121>
60020 PRINT"{CLR}JOHANN DER MASCHINIST{SPA
      CE,DOWN}IST WAHNSINNIG "                    <077>
60030 PRINT"{2DOWN}WAS TUN"                       <086>
60040 PRINT"{2DOWN}1{2SPACE}IHN BERUIHGEN"        <110>
60050 PRINT"{2DOWN}2{2SPACE}IHN SCHLAGEN"         <245>
60060 PRINT"{2DOWN}3{2SPACE}IHN EINSPERREN
      "                                           <113>
60070 ZU=INT(RND(1)*3)+1                          <131>
60080 GET VU                                      <018>
60090 IF VU=1 THEN 60200                          <027>
60100 IF VU=2 THEN 60300                          <105>
60110 IF VU=3 THEN 60400                          <183>
60120 IF VU<>1 OR VU<>2 OR VU<>3 THEN 6008
      0                                           <121>
60130 PRINT A$                                    <105>
60200 IF VU=ZU THEN PRINT"{2DOWN}ER HAT SE
      INEN FEHLER EINGESEHEN":P1=P1+500:LE
      =LE+1:GOSUB 56000:GOTO 315                  <246>
60210 IF VU<>ZU THEN PRINT"{2DOWN}ER SCHLA
      EGT SIE":LE=LE+1:GOSUB 56000:GOTO 31
      5                                           <211>
60300 IF VU=ZU THEN PRINT"{2DOWN}ER IST WI
      EDER NORMAL":P1=P1+500:LE=LE+1:GOSUB
       56000:GOTO 315                             <012>
60310 IF VU<>ZU THEN PRINT"{2DOWN}SIE HABE
      N IHN TOD{5SPACE,DOWN}GESCHLAGEN":LE
      =LU+1:GOSUB 56000:GOTO 315                  <158>
60400 IF VU=ZU THEN PRINT"{2DOWN}ER HAT ZU
      SICH GEFUNDEN":LE=LE+1:P1=P1+1:GOSUB
       56000:GOTO 315                             <073>
```

Listing 2. Hauptprogramm zum Superspiel »Das Boot« (Fortsetzung)

```
60410 IF VU<>ZU THEN PRINT"{2DOWN}ER HAT S
      ICH AUF{7SPACE,DOWN}GEHAENGT":LE=LE+
      1:P1=P1+1:GOSUB 56000:GOTO 315      <003>
61000 GOSUB 63000                         <105>
61020 PRINT"{CLR}SIE TREFFEN MIT EINEM{SPA
      CE,DOWN}DEUTSCHEN U BOOT{6SPACE,DOWN
      }ZUSAMMEN ES HAT KEINEN"            <229>
61025 PRINT"PROVIANT MEHR{9SPACE,DOWN}WIEV
      IEL GEBEN SIE"                      <218>
61040 PRINT"{DOWN}1{2SPACE}3 PROVIANT"    <133>
61050 PRINT"{DOWN}2{2SPACE}5 PROVIANT"    <048>
61060 PRINT"{DOWN}3{2SPACE}GARNICHTS "    <224>
61070 ZU=INT(RND(1)*3)+1                  <115>
61080 GET VU                              <002>
61090 IF VU=1 THEN 61200                  <013>
61100 IF VU=2 THEN 61300                  <091>
61110 IF VU=3 THEN 61400                  <169>
61120 IF VU<>1 OR VU<>2 OR VU<>3 THEN 6108
      0                                   <169>
61130 PRINT A$                            <089>
61200 IF VU=ZU THEN PRINT"{DOWN}DAS WAR HU
      MAN{9SPACE,DOWN}500 PUNKTE":P1=P1+50
      0:LE=LE+1:GOSUB 56000:GOTO 315      <068>
61220 IF VU<>ZU THEN PRINT"{DOWN}SIE ALTER
       GEIZKRAGEN{2SPACE,DOWN}DAFUER GIBTS
       NICHTS":GOSUB 56000:LE=LE+1:GOTO 31
      5                                   <188>
61300 IF VU<>ZU THEN PRINT"{DOWN}DAS U BOO
      T IST WEG{4SPACE,DOWN}KEINE PUNKTE":
      GOSUB 56000:LE=LE+1:GOTO 315        <099>
61310 IF VU=ZU THEN PRINT"{DOWN}DAS WAR SP
      ITZE{8SPACE,DOWN}1000 PUNKTE":LE=LE+
      1:P1=P1+1000:GOSUB 56000:GOTO 315   <185>
61400 IF VU=ZU THEN PRINT"{DOWN}NICHT BESO
      NDERS GUT{3SPACE,DOWN}50 PUNKTE":LE=
      LE+1:P1=P1+50:GOSUB 56000:GOTO 315  <085>
61420 IF VU<>ZU THEN PRINT"{DOWN}SIE SCHWE
      IN{11SPACE,DOWN}KEINE PUNKTE":GOSUB
      56000:LE=LE+1:GOTO 315              <114>
62000 GOSUB 63000                         <089>
62020 PRINT"{CLR}SIE SIND ZU TIEF{6SPACE,D
      OWN}GETAUCHT"                       <219>
62025 PRINT"{DOWN}WAS TUN"                <225>
62040 PRINT"{DOWN}1{2SPACE}WENIG ATMEN"   <241>
62050 PRINT"{DOWN}2{2SPACE}VOLLE KRAFT VOR
      "                                   <250>
62060 PRINT"{DOWN}3{2SPACE}AUFGEBEN{2SPACE
      }"                                  <217>
62070 ZU=INT(RND(1)*3)+1                  <099>
62080 GET VU                              <242>
62090 IF VU=1 THEN 62200                  <255>
62100 IF VU=2 THEN 62300                  <077>
62110 IF VU=3 THEN 62400                  <155>
62120 IF VU<>1 OR VU<>2 OR VU<>3 THEN 6208
      0                                   <217>
62130 PRINT A$                            <073>
62200 IF VU=ZU THEN PRINT"{DOWN}GESCHAFFT{
      13SPACE,DOWN}500 PUNKTE":LE=1:P1=P1+
      500:GOSUB 56000:GOTO 315            <170>
62220 IF VU<>ZU THEN PRINT"{DOWN}SIE SIND
      ERSTICKT{5SPACE,DOWN}DAFUER GIBTS NI
      CHTS":GOSUB 56000:LE=1:GOTO 315     <246>
62300 IF VU<>ZU THEN PRINT"{DOWN}U 96 IST
      EXPLODIERT{3SPACE,DOWN}KEINE PUNKTE"
      :GOSUB 56000:LE=1:GOTO 315          <140>
62310 IF VU=ZU THEN PRINT"{DOWN}SIE SIND A
      UFGETAUCHT{2SPACE,DOWN}1000 PUNKTE":
      LE=1:P1=P1+1000:GOSUB 56000:GOTO 315 <242>
62400 IF VU=ZU THEN PRINT"{DOWN}SIE REALIS
      T{11SPACE,DOWN}200 PUNKTE":LE=1:P1=P
      1+200:GOSUB 56000:GOTO 315          <229>
62420 IF VU<>ZU THEN PRINT"{DOWN}TJA AUFGE
      GEBEN UND WEG{DOWN}KEINE PUNKTE":GOS
      UB 56000:LE=1:GOTO 315              <124>
63000 PRINT"{11DOWN,YELLOW,5RIGHT}CHANCE S
      TAGE{5RIGHT}"                       <009>
63010 POKE QW,6:POKE ZX,151:FOR W=1 TO 10:
      NEXT:POKE QW,0:POKE ZX,0:FOR T=1 TO
      3000:NEXT:RETURN                    <104>
```


Listing 2. Hauptprogramm zum Superspiel »Das Boot« (Schluß)

Drahtseilnerven und Überblick: »Penco«

Ein schnelles Taktik- und Reaktionsspiel in mehreren Akten, für den VC 20 + 8 KByte, mit oder ohne Joystick. Eines der besten Spiele, die es je für den VC 20 zum Abtippen gab.

Heja, da bin ich nun! Wo sind die Monstereier, die Basalt- und Granitsteine? Dort...dort drüben sind zwei Eier nah nebeneinander! Schnell hingelaufen, ein Eisklotz zurechtgeschoben – da schlüpfen sie auch schon aus. Nun schnell den Klotz geschoben. Juhu, beide getroffen! Wo sind die anderen beiden Monster? Ach, weit weg. Was nun? Soll ich versuchen, sie schnell zu treffen, um an den Zeitbonus zu gelangen oder ist es besser, die Granitsteine zusammenzuschieben? Die stehen günstig – also nichs wie los! Ein paar Eisblöcke zerbröseln, ein paar schieben und schon stehen zwei Granitsteine beieinander. Aber was ist das?? Ein Monster kommt! Ich renne schnell weg – aber, verdammt, es verfolgt mich. Schnell zur Bande, die Hochspannungswand aktiviert – da bricht es auch schon betäubt zusammen. So, jetzt überlaufen, oder... oder ist es zu weit weg? Wacht es vorher auf? Nein, ich hab's geschafft! Doch da ist auch schon das letzte Monster, keine drei Eisblöcke mehr entfernt. Schnell hinter diesem Eisblock verstecken. Es kommt direkt auf mich zu, doch da saust ihm auch schon mein Eisblock entgegen und trifft ihn.

Nun ja, in dieser Schwierigkeitsstufe habe ich wohl die Granitsteine nicht zusammenschieben können. Dafür bekomme ich aber, mit nur 18 Sekunden Spieldauer, meinen redlich verdienten Zeitbonus. Puh, ging das wieder schnell her!

Das Programm

Listing 1 zeigt einen speziell für das Maschinensprache-programm entwickelten Monitor. Nach Eingabe dieser Basic-Zeilen in Ihren Computer empfiehlt sich die Speicherung des Monitors auf Kassette.

Das Hauptprogramm liegt im Bereich von \$1400 bis \$2FFF. Um eine Überschreibung durch den Monitor zu vermeiden, muß vor Laden des Monitors jeweils folgende Basic-Zeile im Direktmodus abgearbeitet werden:

```
POKE 44,48 : POKE 48 * 256,0 : NEW
```

Diese Befehlsfolge setzt den Beginn des Basic-Bereichs auf \$3000.

Der Monitor wird mit »RUN« gestartet und zeigt zunächst ein Menü, das heißt eine Überzeile der vorhandenen Optionen. Die Wahl einer »1« bringt den Anwender in die Sektion zur Eingabe von Hexadezimalzahlen. Auf die Eingabe einer Startadresse reagiert der Monitor unter Ausgabe der gewünschten Adresse, gefolgt von einem Ausdruck des momentanen Zelleninhalts. Der Anwender kann nun die Hexadezimalbytes aus Listing 2 eintragen, wobei automatisch auf die nächste Adresse eingestellt wird. Diese Art der

Eingabe spart gegenüber der DATA-Zeilenmethode die Kommata und bei Zahlen über 99 eine dritte Ziffer. Das Drücken der SPACE-Taste überspringt eine Speicherzelle, während die Taste mit dem aufwärts gerichteten Pfeil die vorangegangene Speicherzelle abruft. Der Eingabemodus wird durch ein Drücken der RETURN-Taste verlassen und führt zurück zum Menü. Eine grobe Überprüfung der erfolgten Eingaben auf ihre Richtigkeit erlaubt Sektion 2. Hier wird nach Eingabe einer Blockstartadresse - ein Block besteht aus 256 Byte - von $1400 bis $2F00 ($1400, $1500,...,$1900, $1A00, $1B00,...,$1F00, $2000, $210C,...,$2F00) eine Prüfsumme gebildet, deren Übereinstimmung mit dem vorgegebenen Sollwert anschließend überprüft und gegebenenfalls in Form einer Meldung bestätigt wird. Diese Sektion wird ebenfalls über ein RETURN verlassen. Abschnitt 4 schreibt den Programmbereich von »Penco« auf Band ($1400 - $2F00). Vor dem Betätigen der Aufnahmetasten am Kassettenrecorder sollte beim Anwender allerdings Klarheit über die momentane Position des Tonbands zu dem Schreibkopf bestehen. Der Autor legt deshalb folgendes Verfahren nahe:

1.Initialisierung (einmalig):

a) Basic-Bereich auf $3000 legen (siehe oben)

b) `FOR T = 5120 TO 12287 : POKE T, 0 : NEXT`

eingeben und über RETURN abarbeiten. Diese Zeile schreibt Nullen in den Maschinensprachebereich und zeigt dem Anwender später an, wo er das Übertragen der Bytes aus Listing 2 fortzusetzen hat.

c) Laden des Monitors uns Starten desselben (danach STOP-Taste am Recorder drücken!)

d) Speichern des Hauptprogrammbereichs mit Punkt 4.

Normalablauf:

a) Basic-Bereich auf $3000 legen

b) Monitor laden und starten

c) Laden des Hauptprogramms über Taste 3

d) Rückspulen des Bands auf den Monitorbeginn und Eingabe eines »VERIFY«-Kommandos. Das Band ist nun in der richtigen Position für eine Aufzeichnung des ergänzten Hauptprogramms. Nach der Eingabe Ihres Tagespensums an Hex-Bytes und Überprüfung der Werte in Sektion 2 hält ein Abrufen der Sektion 4 die Ergebnisse elektromagnetisch fest.

Eines schönen Tages, nach Eingabe von lediglich 7168 Byte, führt schließlich auch für Sie das Abrufen der Taste 5 im Menü des Monitors nicht zu einem Absturz des Systems, sondern sollte das Hauptprogramm »Penco« starten.

Penco in Aktion

Das Titelbild des Hauptprogramms zeigt unseren Hauptdarsteller, den kleinen Pinguin Penco, in Aktion. Das Zentrieren des Bildes ist über Joystick oder die Tasten »/« und »X« möglich. Gestartet wird mit F1 oder dem Feuerknopf. Anzumerken ist die Möglichkeit, das Spiel mit Hilfe der Funktionstaste F7 jederzeit zu unterbrechen beziehungsweise mit F5 zur Titelseite zurückzukehren.

Das Aktionsfeld des Pinguins wird eingegrenzt durch ein Rechteck aus Hochspannungswänden und ist mit einer Anzahl von blauen, abbaubaren Basaltsteinen angefüllt. Daneben beleben drei Granitsteine, mehrere Monstereier und farbig unterlegte Basaltsteine die Szene.

Der Spieler hat, vertreten durch den Pinguin, der zu Beginn jeder Ebene in der Mitte der Spielfläche erscheint, folgende Optionen:

Er kann Blöcke in eine der vier Richtungen verschieben oder zerbröseln, je nachdem, ob sich in der Bewegungsrichtung ein weiterer Block in direkter Nachbarschaft befindet. Die angeschobenen Blöcke rutschen so weit, bis sie auf ein solides Hindernis treffen, wobei sie unterwegs eventuell Monster zermalmen. Eine weitere wichtige Möglichkeit hat der Spieler in dem Aufwellen der Spielfeldumrandung. Beim Zurückschwingen der Wandung in die Ausgangsposition werden alle Monster in direkter Nachbarschaft betäubt. In dieser Form können sie vom Spieler einfach überlaufen werden. Verliert der Spieler ein Leben, so werden die Monster in den Ecken des Spielfeldes neu gestartet, wobei dort befindliche Granite eventuell ausgelöscht werden. Gleiches gilt für die Startposition des Pinguins.

Bewegung des Spielers

Der Pinguin wird über die Tastatur oder alternativ mit einem Joystick kontrolliert.

»Z«, »X« - horizontal
»;«, »/« - vertikal
SHIFT (mit Richtung) - schieben

Das für das Spiel charakteristische Schieben und Zerstören der Basaltblöcke erfordert ein exaktes Aufstellen neben, über oder unter den Quadern. Um einen Block zu manipulieren, muß gleichzeitig zum Drücken der SHIFT-Taste beziehungsweise des Feuerknopfes die gewünschte Richtung eingeschlagen werden.

Die Koordination der Bewegung wird zunächst wahrscheinlich Schwierigkeiten bereiten, aber nicht den Mut verlieren. Die Steuerung ist trainierbar und läßt sich durchaus beherrschen.

Die Rolle der Monster

Die nach einiger Zeit aus den Eiern schlüpfenden Monster setzen sich sofort in Bewegung, zerstören manche Blöcke und suchen den Pinguin, wobei ihre Bewegungen zunächst sehr zufällig sind. Bei Kontakt von Pinguin und Monster verliert der Spieler ein Leben. Die Monster können entweder im Normalzustand mit bewegten Blöcken weggewischt werden, oder nach erfolgter Betäubung bei Aufwellung der Wände durch ein Überlaufen von der Bildfläche verschwinden. Jedes ausgelöschte Monster erhöht den Punktestand um 80 beziehungsweise 40 Punkte. Sollte nur noch ein Monster umherlaufen, so wird dieses versuchen, seine »Heimat«-Ecke anzusteuern und dort in ein Ei zurückriechen. Dieses letzte Monster kann nicht betäubt werden!

Die Granitsteine und Bonusbasalte

Es gibt auf dem Spielfeld drei Granitsteine, die sich gestaltlich abheben. Werden diese Steine in eine geschlossene vertikale oder horizontale Reihe gebracht, so gibt es einen Bonus von 10, 100, 500 oder 1000 Punkten. Gleichzeitig werden die Monster für einen kurzen Zeitraum betäubt. Für das Zerstören eines farbig unterlegten Bonusbasaltes gibt es 50 Punkte. Die Granitsteine sind unzerstörbar.

Die Anzeige

Angezeigt wird in der Kopfzeile das jeweilige Bestergebnis (HIGH), der aktuelle Punktestand (SCORE), die Anzahl der Leben und die aktuelle Spielebene (links beziehungsweise rechts vom Herz). Alle 10000 Punkte wird die Anzahl der Leben um eins erhöht, alle 10 Ebenen die Logik der Monster verschärft.

Und nun viel Spaß! (Der Highscore liegt übrigens bei 44440, Ebene 31....)

(Frank Goroncy/aa)

```
2 REM PENCO:MONITOR                                                      <243>
4 REM VC-20 MIT MINDESTENS 8 KBYTE ERWEITERUNG                           <173>
6 REM FRANK GORONCY/WIESENWEG 21/3250 HAMELN 13                          <105>
10 REM .PRUEFSUMMEN                                                      <185>
12 DIM CS(27):FOR T=0 TO 27:READ CS(T):NEXT:DATA 13017.17618,10874,20856 <072>
14 DATA 36576,27458,11408,25631,20224,30214,28582,28081,31135,26987,2683
   2,28356                                                               <115>
16 DATA 28299,19691,24254,23699,20732,24057,17151,29295,28710,24105,1544
   4,24794                                                               <127>
18 REM .MENUE                                                            <191>
20 HX$="0123456789ABCDEF":PRINT"{CLR,2DOWN,2SPACE}EINGABE{9SPACE}1"      <077>
22 PRINT"{DOWN,2SPACE}UEBERPRUEFUNG{3SPACE}2"                            <099>
24 PRINT"{DOWN,2SPACE}LADEN{11SPACE}3"                                   <089>
26 PRINT"{DOWN,2SPACE}ABSPEICHERN{5SPACE}4"                              <123>
28 PRINT"{DOWN,2SPACE}'PENCO' STARTEN 5{DOWN}"                           <213>
30 POKE 198,0:WAIT 198,1:GET A$:IF A$<"1"OR A$>"5"THEN 30                <190>
40 ON VAL(A$)GOSUB 100,400,200,202,300:GOTO 20                           <200>
98 REM .EINGABE                                                          <158>
100 AD$="":INPUT"{DOWN,2SPACE}STARTADRESSE";AD$:GOSUB 502:LO=AD          <213>
102 PRINT:C=PEEK(LO):AD=LO:GOSUB 504:PRINT"{2SPACE}"AD$" ";:B=C:GOSUB 50
    6:PRINT B$" ";                                                       <157>
104 GOSUB 508:IF A$=CHR$(13)THEN RETURN                                  <001>
106 IF A$=" "THEN LO=LO+1:GOTO 102                                       <136>
108 IF A$="↑"THEN LO=LO-1:GOTO 102                                       <008>
110 B$=A$:GOSUB 508:B$=B$+A$:IF A$=CHR$(13)OR A$=" "OR A$="↑"THEN GOTO 1
    02                                                                   <226>
112 GOSUB 500:POKE LO,B:LO=LO+1:GOTO 102                                 <060>
198 REM .LADEN & ABSPEICHERN                                             <087>
200 GOSUB 204:POKE 780,0:SYS 65493:RETURN                                <211>
202 GOSUB 204:POKE 193,0:POKE 194,20:POKE 174,0:POKE 175,48:SYS 63106:RE
    TURN                                                                 <219>
204 POKE 185,1:POKE 186,1:POKE 183,0:POKE 144,0:POKE 187,0:RETURN        <045>
298 REM .'PENCO' STARTEN                                                 <242>
300 SYS 9508                                                             <216>
398 REM .UEBERPRUEFUNG                                                   <050>
400 AD$="":INPUT"{DOWN,2SPACE}BLOCKADRESSE";AD$:IF AD$=""THEN RETURN     <238>
402 GOSUB 502:B=AD/256-20:IF B<>INT(B)OR B<0 OR B>27 THEN 400            <027>
404 PRINT"{2SPACE}SUMME";:C=0:FOR T=AD TO AD+255:C=C+PEEK(T):NEXT:IF CS(
    B)<>C THEN PRINT"{SPACE,RVSON}NICHT";                                <242>
406 PRINT" I.O.":GOTO 400                                                <184>
498 REM .HEX/DEZ-ROUTINEN                                                <021>
500 L=ASC(LEFT$(B$,1)):R=ASC(RIGHT$(B$,1)):B=(L-48+(L>57)*7)*16+R-48+(R>
    57)*7:RETURN                                                         <164>
502 B$=LEFT$(AD$,2):GOSUB 500:AD=B*256:B$=RIGHT$(AD$,2):GOSUB 500:AD=AD+
    B:RETURN                                                             <227>
504 B=INT(AD/256):GOSUB 506:AD$=B$:B=AD-B*256:GOSUB 506:AD$=AD$+B$:RETUR
    N                                                                    <022>
506 B$=MID$(HX$,B/16+1,1)+MID$(HX$,B-INT(B/16)*16+1,1):RETURN            <236>
508 POKE 198,0:WAIT 198,1:GET A$:IF A$=CHR$(13)OR A$=" "OR A$="↑"THEN RE
    TURN                                                                 <079>
510 IF A$<"0"OR A$>"F"OR A$<"A"AND A$>"9"THEN 508                        <156>
512 PRINT A$;:RETURN                                                     <242>
```


Listing 1. Der Monitor zu »Penco«. Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
13F4 EA EA EA EA EA EA EA EA EA EA EA EA 00 00 00 00 00 00 00 00 00 08 34 1C 1C
140D 14 68 58 14 0E 3E 3E 1C 00 14 2C 00 00 00 00 00 08 34 1C 1C 14 68 58 14 0E
1426 3E 3E 1C 00 14 2C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
143F 00 00 00 00 00 00 00 00 00 00 10 2C 38 38 44 D6 BA 38 7C 7C 7C 38 00 28 6C
1458 00 10 2C 38 38 28 16 1A 28 70 7C 7C 38 04 6C 40 00 10 2C 38 38 28 16 1A 28
1471 70 7C 7C 38 00 28 34 00 08 34 1C 1C 14 68 58 14 0E 3E 3E 1C 40 66 02 00 08
148A 34 1C 1C 14 68 58 14 0E 3E 3E 1C 00 14 2C 00 10 38 38 38 44 D6 BA 38 7C 7C
14A3 7C 30 08 2C 60 00 10 38 38 38 44 D6 BA 38 7C 7C 7C 18 20 68 0C 00 08 16 1C
14BC 1C 0B 0D 14 1B 1F 1F 1E 0C 00 14 1A 00 10 68 38 38 D0 B0 28 D8 F8 F8 78 30
14D5 00 28 58 00 00 00 10 38 38 38 44 D6 BA 38 7C 7C 7C BA C6 38 38 38 44 D6 BA
14EE 38 7C 7C 7C 38 00 28 28 28 28 00 18 2C 5A 7E 38 0C 22 00 01 02 05 07 03 01
1507 02 00 80 C0 A0 E0 80 00 40 00 18 34 5A 7E 1C 30 44 00 80 40 A0 E0 C0 80 40
1520 00 01 03 05 07 01 00 02 00 18 24 5A 6E 3C 58 02 76 3C 1A 40 00 00 00 00 00
1539 00 00 00 00 18 24 5A 00 18 3C 5A 7E 24 58 02 00 00 00 00 00 18 3C 5A 7E 3C
1552 1A 40 00 00 00 00 00 10 40 02 18 3C 5A 7E 08 20 02 00 18 3C 5A 7E 00 00 00
156B 00 00 00 00 00 00 00 00 00 00 00 00 00 18 3C 5A 7E 24 18 42 00 FE 4C F6 7E
1584 9A 7C BA D4 AE 44 E2 54 8A 30 96 00 B4 08 56 50 2A 40 AA 54 00 44 82 54 02
```

Listing 2. Hauptprogramm von »Penco«. Bitte nur mit dem Monitor aus Listing 1 eingeben.

```
159D 20 94 00 44 00 10 40 08 00 22 40 00 00 00 54 02 20 94 00 FE 7C FE 7C BA 7C
15B6 BA 54 AA 44 82 44 82 00 82 00 FE FE FE 82 FE 82 FE 00 00 7C 00 7C 00 00 00
15CF 00 10 38 38 7C 7C 7C FC FC FC F8 FA 52 20 44 28 00 10 38 38 7C 7C 3C 50 EC
15E8 FE F8 5A 12 28 44 28 00 10 28 38 58 74 18 50 2C DE 7C 5A 7C 28 44 28 00 00
1601 20 00 58 24 10 00 2C 5A 3C 5A 7E 24 74 28 00 00 20 00 08 00 10 00 40 58 3C
161A 5A 7E 24 58 22 00 38 44 38 00 10 38 38 BA C6 54 38 7C 7C 7C BA C6 00 00 00
1633 00 00 02 05 02 05 02 05 02 05 02 05 02 05 02 05 00 00 00 00 00 55 AA 55 00
164C 00 00 00 00 40 A0 40 00 00 00 00 00 40 A0 40 A0 40 A0 40 A0 00 00 00 00 00
1665 A0 40 A0 00 00 00 00 00 AA 55 AA 05 02 05 04 0A 04 0A 04 0A 04 0A 04 0A 02
167E 05 02 00 00 00 05 12 6D 90 40 00 00 00 50 A4 59 06 01 A0 40 A0 20 50 20 50
1697 20 50 20 50 20 50 40 A0 40 40 90 6D 12 05 00 00 00 01 06 59 A4 50 00 00 00
16B0 3C 00 02 3C 00 00 40 00 3E 00 00 38 00 00 7E 00 00 20 10 08 04 00 42 00 1C
16C9 22 00 0C 40 22 1C 00 18 24 00 00 42 24 18 00 00 00 00 00 00 00 00 00 10 80
16E2 04 08 2A 2A 2A 2A 00 36 7F 7F 7F 3E 1C 08 00 44 44 66 7E 66 66 66 00 10 30
16FB 30 38 38 38 38 00 7E 66 60 6E 66 72 7E 00 7E 66 66 60 60 62 7E 00 18 24 46
1714 46 46 6E 3C 00 7C 64 64 7E 66 65 66 00 7E 60 60 7C 70 70 7E 00 7E 36 30 38
172D 38 38 38 00 6E 56 46 46 46 76 76 00 7C 64 64 7E 66 66 7E 00 62 72 6A 66 62
1746 6A 6A 00 42 46 46 46 4E 6E 3C 00 00 00 18 3C 18 00 00 00 20 20 20 60 60 70
175F 7E 00 3C 24 24 7E 66 66 66 00 46 46 46 46 46 2C 18 FF FF FF FF FF FF FF FF
1778 00 7E 66 66 7E 70 70 70 00 7E 42 42 42 42 42 7E 00 08 18 28 08 08 08 3E 00
1791 7E 42 02 7E 40 42 7E 00 7E 42 02 3E 02 42 7E 00 08 18 28 48 7E 08 08 00 7E
17AA 42 40 7E 02 42 7E 00 7E 42 40 7E 42 42 7E 00 7E 42 02 04 08 10 20 00 7E 42
17C3 42 7E 42 42 7E 00 7E 42 42 7E 02 42 7E 00 00 77 51 57 54 77 00 69 8A 48 A8
17DC C8 B8 E8 D8 31 43 45 F1 41 41 41 41 00 7E 46 40 7E 06 46 7E 00 10 38 FE 7C
17F5 38 6C 44 59 7A 48 98 B8 C9 D8 E8 0A 2A 2A AA AA AB AB AB A0 A8 A8 AA AA EA
180E FF FF 02 02 02 02 02 C2 CA FA AA AA AA AA AA AA AA AA 2A 2A 2A EA EA FA FE
1827 FB 00 C0 C0 F0 FA FA FA EA 00 00 00 00 A8 A8 AA AA 00 00 00 00 2A 2A 2A 2A
1840 00 00 00 00 28 EB AB AB 00 00 00 00 00 00 00 C0 AA AA AA AA AA AB AB AB AF
1859 AB AB AA AA EA EA EA FA FA FA EA EA EA EA EA EA EA EA EA AA AA AA AA FF F3
1872 F3 F0 A0 AC AC AF EA EA 2A 2A 2A 2A 2A 2A FA FA AA AA AA AA FF FF EA EA EA
188B EA EA EA EA EA AB AB BF FF FF FF FF FC C0 C0 C0 C0 C0 C0 C0 00 55 55 15 2A
18A4 2A 0F 0F 0F 55 55 57 AB AB FE FE FF D5 D5 FF FF FF C0 C0 C0 55 55 D5 EA EA
18BD 0F 0F 0F 5F 5F FF FF FF F0 F0 F0 15 15 05 0A 0A 03 03 03 55 55 55 AA AA FF
18D6 FF FF D5 D5 D5 EA EA FF FF CF F0 F0 F0 F0 F0 F0 F0 F0 00 00 00 00 00 00 00
18EF 00 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 20 FE 1D A0 0A A9 0F 20
1908 2C 19 A6 3F A9 01 E0 80 D0 09 18 6D 00 90 29 9F 8D 00 90 E0 08 D0 09 18 6D
1921 01 90 29 3F 8D 01 90 88 D0 DA 60 85 40 98 48 A0 F0 88 D0 FD C6 40 D0 F7 68
193A A8 60 A2 47 A9 08 8D 0F 90 A9 00 8D 02 90 2C A2 4E 2C A2 74 BC 00 29 F0 0D
1953 8A 48 98 A0 28 20 1E CB 68 AA E8 D0 EE A9 12 8D 02 90 60 A9 38 8D 03 90 A9
196C CD 8D 05 90 A9 F6 8D 0F 90 A2 00 20 43 19 20 9A 29 4C B2 29 A9 0B 85 40 A0
1985 D5 A2 0F A9 00 99 37 10 99 0F 11 A9 06 99 37 94 99 0F 95 88 CA 10 EC 88 88
199E C6 40 10 E4 60 30 0A 18 65 61 90 02 E6 62 4C B6 19 38 65 61 B0 02 C6 62 85
19B7 61 60 A9 F2 A2 10 85 61 86 62 20 25 1B 29 03 A8 B9 FC 19 AA BD 00 1C A8 4A
19D0 4A 4A 85 69 B0 24 98 29 03 A8 86 41 BE FC 28 8A 20 A3 19 A0 00 98 91 61 A0
19E9 12 91 61 A9 25 20 2C 19 C6 69 10 EA A6 41 E8 D0 D1 60 2A 00 1C 39 51 2A 2A
1A02 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A1B 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A34 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A4D 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A66 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A7F 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1A98 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1AB1 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
1ACA 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 8D 69 45 21 FD D9 B5 6D 11 11 11 11
1AE3 10 10 10 10 70 60 50 40 30 20 10 00 27 27 27 27 27 27 27 27 2A 2A 2A 2A 2A
1AFC 2A 2A 2A 2A 8A 4A 85 40 98 4A A9 B9 4A 1B 0A 48 A9 10 69 00 85 62 09 84 85
1B15 64 68 18 65 40 90 04 E6 62 E6 64 85 61 85 63 60 38 A5 09 65 0C 65 0D 85 08
1B2E A2 04 B5 08 95 09 CA 10 F9 60 48 A6 0E A4 0F 20 00 1B 68 A8 B1 61 C9 2D 90
1B47 01 38 60 12 1B 24 2D 36 3F 48 51 5A 63 6C 75 7E 87 90 99 A2 AB B4 BD C6 CF
1B60 D8 E1 20 25 1B 29 3E A8 20 25 1B 29 3E C9 22 B0 F7 AA 20 00 1B A0 00 B1 61
1B79 C9 36 D0 E5 B1 63 29 07 C5 06 F0 DD 60 20 B3 27 20 AC 24 20 49 19 20 B9 19
1B92 A9 02 85 07 20 62 1B A9 38 91 61 A0 12 A9 39 91 61 A5 06 91 63 A0 00 91 63
1BAB C6 07 10 E7 A9 05 85 07 20 62 1B A5 06 91 63 A0 12 91 63 C6 07 10 F1 60 2A
1BC4 2A A2 00 B5 00 D5 03 90 10 D0 05 E8 E0 03 D0 F3 A2 02 B5 00 95 03 CA 10 F9
1BDD 20 F3 1C 20 AC 24 A9 00 85 8E A9 F8 85 C3 A9 E0 85 C4 A9 73 8D 14 10 A9 01
1BF6 85 8F 60 25 24 13 12 01 00 2A 08 13 21 0B 50 0A 19 0A 18 32 09 1B 11 0A 00
1C0F 0A 11 1B 29 0A 18 0A 29 53 08 22 20 FF 1A 09 0A 28 13 09 0B 18 22 08 33 19
1C28 0B 18 0B 31 12 09 13 29 52 10 1B 01 1B 08 0A 10 FF 09 1A 09 33 28 0B 41 52
```

```
1C41 30 1B 28 0F 11 0A 20 53 19 0A 08 0A 21 02 01 FF 13 08 0B 19 12 11 13 09 52
1C5A 28 0B 11 1B 08 0A 10 1A 30 0B 21 0B 20 33 11 22 21 FF 78 20 C2 2E A9 7F 8D
1C73 22 91 AD 20 91 A2 FF 8E 22 91 29 80 85 3F AD 1F 91 29 3C 05 3F 49 BC 85 3F
1C8C 20 1E EB A5 C5 C9 40 F0 50 C9 1E D0 02 A9 08 C9 16 D0 02 A9 04 C9 1A D0 02
1CA5 A9 80 C9 21 D0 02 A9 10 AC 8D 02 F0 02 09 20 85 3F C9 3F D0 1C AD 0E 90 48
1CBE A9 00 8D 0E 90 20 E8 1C F0 FB 20 E8 1C D0 FB 20 E8 1C F0 FB 68 8D 0E 90 C9
1CD7 37 D0 0B BA A9 25 9D 06 01 A9 2C 9D 05 01 4C 15 EB 20 1E EB A5 C5 C9 3F 60
1CF0 A2 05 2C A2 02 A9 00 95 00 CA 10 F9 60 A4 5F C0 98 D0 06 E6 0F A0 A8 D0 26
1D09 C0 99 D0 04 C6 0F D0 F4 C0 78 D0 06 C6 0E A0 88 D0 14 C0 58 D0 06 E6 0E A0
1D22 68 D0 0A 95 3F F0 04 85 69 D0 17 A0 48 84 D2 98 29 FE A8 A2 00 B9 00 14 9D
1D3B 08 14 C8 E8 E0 10 D0 F4 60 29 20 F0 09 A5 69 49 20 F0 03 4C B2 1F A5 69 49
1D54 08 D0 11 E6 0F A9 24 20 38 1B 90 04 C6 0F D0 C9 A0 98 D0 C7 A5 69 49 04 D0
1D6D 11 C6 0F A9 00 20 38 1B 90 04 E6 0F D0 B2 A0 99 D0 B0 A5 69 29 10 F0 18 C6
1D86 0E A9 00 29 38 1B B0 07 99 12 20 38 1B 90 04 E6 0E D0 94 A0 78 D0 92 A5 69
1D9F 29 80 F0 A0 E6 0E A9 01 20 38 1B B0 07 A9 13 20 38 1B 90 04 C6 0E D0 E0 A0
1DB8 58 D0 E0 B1 63 C9 38 90 03 68 68 60 20 BA 27 B5 A2 D0 0D A5 63 95 97 A5 64
1DD1 95 50 A9 2E 95 A2 60 CA 10 EC 60 A5 64 09 84 85 64 B1 63 29 07 C5 06 D0 F1
1DEA A9 50 4C 99 24 A5 FE D0 E8 86 FE 20 2F 1D A9 45 85 94 EA 60 A5 D2 C5 5F F0
1E03 F9 85 5F A2 06 BC F8 1B B1 FB C9 09 B0 04 A9 00 91 FB CA D0 F0 A0 30 A2 00
1E1C 8A 9D 18 14 E8 88 D0 F9 A6 0E A4 0F 20 00 1B A0 08 A9 10 85 6A A5 0E A2 5E
1E35 29 01 F0 02 A2 52 8E 50 1E A2 00 A5 0F 29 01 F0 02 A2 04 A9 00 85 6B B9 00
1E4E 14 4C 5E 1E 4A 66 6B 4A 66 6B 4A 66 6B 4A 66 6B 9D 18 14 A5 6B 9D 30 14 E8
1E67 C8 C6 6A D0 DC A9 18 85 63 A9 14 85 64 A5 62 85 FC 09 84 85 68 A5 61 85 67
1E80 85 FB A9 00 85 6F 20 B4 1E 20 B4 1E 20 B4 1E A5 FC 85 62 09 84 85 68 A5 FB
1E99 18 69 01 90 04 E6 68 E6 62 85 61 85 67 20 B4 1E 20 B4 1E 20 B4 1E A5 6F D0
1EB2 5C 60 A0 00 84 6E 84 66 B1 61 F0 2F 85 69 0A 26 66 0A 26 66 0A 85 65 A9 14
1ECB 65 66 85 66 B1 63 F0 0B E6 6E 31 65 F0 03 20 D4 2F B1 63 11 65 91 63 C8 C0
1EE4 08 D0 E8 A0 00 A5 6E F0 0B A5 63 4A 4A 4A 91 61 A5 96 91 67 A5 63 18 69 08
1EFD 85 63 A5 61 18 69 12 90 04 E6 62 E6 68 85 61 85 67 60 20 9F 27 A2 07 B5 2B
1F16 F0 30 C9 05 F0 2C C9 01 F0 28 A8 B5 42 85 61 B5 73 85 62 C0 06 D0 07 A9 36
1F2F 20 09 2B D0 14 A0 00 98 91 61 B5 83 10 0B B5 7B 20 A3 19 98 91 61 EA EA EA
1F48 CA 10 C9 A2 0F BD 20 16 9D 08 14 CA 10 F7 A5 06 85 96 20 06 1E A9 01 85 96
1F61 A2 04 A9 FF 20 2C 19 CA 10 F8 CE 14 10 AD 14 10 C9 70 D0 03 4C 5C 22 A2 0F
1F7A A9 00 9D 08 14 CA 10 FA 20 06 1E 4C 9E 22 A2 08 B4 A2 F0 20 C8 C8 C0 36 90
1F93 02 A0 00 94 A2 B5 97 85 61 B5 50 85 62 98 A0 00 91 61 F0 03 18 69 01 A0 12
1FAC 91 61 CA 10 D9 60 A5 69 A6 0E A4 0F E0 02 D0 0C 2C E0 16 F0 07 A0 C8 A2 07
1FC5 4C EF 1D E0 20 D0 0C 2C E1 16 F0 07 A0 B8 A2 0F 4C EF 1D C0 02 D0 0C 2C E2
1FDE 16 F0 07 A0 E8 A2 0B 4C EF 1D C0 2E D0 0C 2C E3 16 F0 07 A0 D8 A2 13 4C EF
1FF7 1D 4C A6 20 A0 00 A5 69 29 9C C9 90 D0 04 A2 01 A0 B8 C9 10 D0 04 A2 FE A0
2010 C8 C9 08 D0 04 A2 24 A0 D8 C9 04 D0 04 A2 DB A0 E8 98 F0 5B 86 69 20 2F 1D
2029 A5 69 20 A3 19 A0 00 B1 61 C9 36 90 49 C9 3A B0 45 A5 61 85 63 A5 62 85 64
2042 A5 69 20 A3 19 EA EA B1 61 C9 36 90 03 4C 93 22 A2 03 B5 BA D0 24 A5 69 95
205B BA A5 63 95 AE A5 64 95 B2 A0 00 B1 63 C9 36 D0 14 A5 64 09 84 85 64 A9 06
2074 91 63 A0 12 91 63 D0 04 CA 10 D5 60 4C EE 27 EA EA EA A5 0F 29 02 F0 F3 20
208D 00 1B 4C FB 1F A9 00 20 F2 26 B5 B6 F0 0C A8 A9 80 20 99 24 88 D0 F8 98 95
20A6 B6 A0 00 B1 65 C9 38 D0 5A 84 8C 84 8D A5 66 85 62 A5 65 38 E9 6C B0 02 C6
20BF 62 85 61 A2 04 A9 24 20 A3 19 B1 61 C9 38 18 D0 01 38 26 8C CA 10 EE A5 66
20D8 85 62 A5 65 38 E9 03 B0 02 C6 62 85 61 A2 04 A9 01 20 A3 19 B1 61 C9 38 18
20F1 D0 01 38 26 8D CA 10 EE A2 02 BD FD 23 C5 8C F0 08 C5 8D F0 04 CA 10 F2 60
210A A5 A0 85 BE A5 A1 85 BF 20 5E 21 20 91 21 20 4C 19 A5 BF 85 A1 A5 BE 85 A0
2123 20 F9 26 60 A9 07 95 2B A9 05 95 83 A9 20 95 37 60 98 0A 0A 0A 85 61 A9 14
213C 85 62 A5 70 69 02 0A 0A 0A 85 65 A9 00 69 14 85 66 A0 07 B1 61 F0 07 31 65
2155 F0 03 4C 06 1E 88 10 F2 60 20 91 27 A2 07 B5 2B C9 02 D0 24 B5 73 85 62 B5
216E 42 85 61 A9 04 95 2B A9 1A 95 37 A9 2B 91 61 B5 83 10 0C 29 7F 95 83 B5 7B
2187 20 A3 19 98 91 61 CA 10 D3 60 A0 00 B9 24 10 C9 46 90 05 A9 7E 99 24 10 B9
21A0 00 11 C9 46 90 05 A9 7E 99 00 11 C8 D0 E5 A0 05 B9 0A 27 99 EA 11 88 10 F7
21B9 20 25 1B 29 03 A8 B9 54 22 85 66 B9 58 22 85 67 20 45 22 A2 02 A9 A0 20 2C
21D2 19 CA 10 F8 A0 00 B9 24 10 C9 7E D0 0A 20 25 1B 29 07 F0 F9 99 24 94 B9 00
21EB 11 C9 7E D0 0A 20 25 1B 29 07 F0 F9 99 00 95 C8 D0 DB F8 A5 67 38 E9 01 85
2204 67 A5 66 E9 00 85 66 D8 20 45 22 A9 01 20 99 24 20 AC 24 A9 4C 85 5D 85 5C
221D A5 67 05 66 D0 B3 A2 00 BD 24 10 C9 7E D0 05 A9 01 9D 24 94 BD 00 11 C9 7E
2236 D0 05 A9 01 9D 00 95 CA D0 E5 A5 94 D0 FC 60 A0 03 A2 67 A9 F0 85 61 A9 11
224F 85 62 4C C3 24 10 05 01 00 00 00 00 10 20 C1 27 68 68 A0 00 A9 00 99 ED 10
2268 99 23 11 99 11 11 B9 E6 26 99 FF 10 A9 03 99 FF 94 A9 50 20 2C 19 C8 C0 0C
2281 D0 E0 20 91 27 A2 0A A9 FF 20 2C 19 CA 10 F8 4C 34 25 20 BB 1D 4C DC 1D A0
229A 00 4C BB 1D A0 03 84 6A A2 07 B5 2B C9 06 D0 04 A9 01 95 2B C9 04 D0 04 A9
22B3 02 95 2B C9 03 D0 05 48 48 4C DA 2E C9 02 D0 21 A4 6A 10 08 A9 05 95 2B 95
22CC 37 D0 15 B9 6C 2A 85 62 09 84 85 64 B9 70 2A 85 61 85 63 20 C5 2A C6 6A CA
```

```
22E5 10 BD A2 06 A9 FF 20 2C 19 CA 10 F8 68 68 A5 27 85 A0 A5 28 85 A1 4C 53 25
22FE 2A 2A A5 FE D0 01 60 29 1C 4A 4A AA BD CD 23 85 69 0A 85 F9 BD E5 23 85 F7
2317 BD B9 23 85 4E BD B5 23 85 4F 85 F8 BD D1 23 85 60 A6 FE A0 00 BD BA 23 91
2330 4E A5 60 A8 BD D2 23 91 4E A5 60 0A 65 4E 90 02 E6 4F 85 4E C6 69 D0 E2 8A
2349 29 03 F0 03 C6 FE 60 A0 00 B1 F7 C9 1F 90 07 C9 2B B0 03 20 71 23 A5 F7 18
2362 65 60 90 02 E6 F8 85 F7 C6 F9 D0 E4 84 FE 60 A2 07 B5 2B C9 02 D0 39 B5 73
237B 85 62 C5 F8 D0 31 B5 42 85 61 C5 F7 F0 09 18 75 7B C5 F7 D0 22 95 42 A9 04
2394 95 2B A9 1A 95 37 B5 83 10 0F 29 7F 95 83 98 91 61 B5 7B 20 A3 19 98 91 61
23AD A9 2B 91 F7 60 CA 10 BE 60 10 10 10 11 36 25 47 E7 47 4F 47 4E 4D 50 4D 51
23C6 4B 52 4B 53 49 54 49 55 0C 08 0C 08 12 01 12 01 47 4E 47 4F 4D 51 4D 50 4B
23DF 53 4B 52 49 55 49 54 37 37 46 D5 F8 18 65 02 85 02 A9 00 65 01 85 01 A9 00
23F8 65 00 85 00 60 0E 1C 07 F6 B6 A5 65 85 63 A5 66 09 84 85 64 A0 00 A5 06 91
2411 63 A0 12 91 63 60 A9 00 95 BA 8A 48 20 92 20 68 AA 4C 95 24 A2 03 B4 BA F0
242A 6A B5 AE 85 65 85 61 B5 B2 85 66 85 62 98 20 A3 19 A0 00 B1 61 C9 30 B0 D4
2443 C9 1F 90 93 20 00 24 A0 12 B1 61 C9 30 B0 C5 C9 1F 90 0B C9 26 F0 07 C9 2A
245C F0 03 20 00 24 A0 00 B1 65 85 69 98 A0 12 91 65 A8 91 65 A5 61 85 63 95 AE
2475 A5 62 95 B2 09 84 85 64 A5 69 91 61 A5 66 09 84 85 66 B1 65 91 63 A0 12 91
248E 63 E6 69 A5 69 91 61 CA 10 8F 60 20 EA 23 A5 00 C5 8B 90 08 18 69 01 85 8B
24A7 EE 14 10 D8 60 A2 05 A9 0C 85 61 A9 10 85 62 20 C1 24 A9 1E 85 61 20 C1 24
24C0 60 A0 04 B5 00 29 0F 18 69 70 91 61 88 10 02 CA 60 B5 00 29 F0 4A 4A 4A 4A
24D9 18 69 70 91 61 CA 88 10 E1 60 A2 03 D6 4A 10 25 B4 92 F6 92 B5 5B 85 35 A9
24F2 26 85 36 B1 35 D0 02 95 92 48 29 0F 0A EA 95 4A 68 4A 4A 4A 4A A8 B9 14 25
250B 9D 0A 90 CA 10 D4 4C 6C 1C 00 AF B7 BF C3 C9 CF D1 D7 DB DF E1 E4 E7 E8 EB
2524 A9 0F 8D 0E 90 20 F0 1C A2 FF 78 9A 58 20 80 27 20 91 27 20 C5 1B 20 66 19
253D 20 3C 19 A9 00 85 00 20 D2 27 20 7C 2A 20 BD 26 20 61 26 20 86 1B A9 10 85
2556 0E A9 16 85 0F A9 00 85 3F 85 5F 8D F2 10 8D 04 11 20 DC 27 20 FD 1C 20 FE
256F 1D A2 3C A0 FF 88 D0 FD CA D0 F8 20 25 24 20 88 1F 20 AC 24 20 A2 2A 20 00
2588 23 20 66 2D 4C 6A 25 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
25A1 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
25BA 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
25D3 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
25EC 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 00 06 82 92 A8
2605 84 A4 B4 A4 94 84 9A 54 86 52 94 A4 88 54 82 A2 94 58 62 72 8C 00 0C 34 08
261E 14 08 34 0C 0C 34 08 14 08 34 0C 0C 74 08 44 08 74 0C 0C 74 08 44 08 74 0C
2637 00 F1 D1 72 24 00 11 31 F1 C4 00 70 40 00 F1 E1 F1 E1 F1 E1 00 44 78 94 88
2650 A4 68 84 78 00 44 64 94 88 74 68 00 F0 C0 10 60 00 A5 8E 29 0F 20 7E 26 A9
2669 00 20 95 26 A5 8E 29 F0 4A 4A 4A 4A 20 7E 26 A9 00 20 95 26 60 85 07 0A 18
2682 65 07 A8 A9 02 85 07 B9 9F 26 20 95 26 C8 C6 07 10 F5 60 A2 07 0A 7E D0 17
269B CA 10 F9 60 7C 44 7C 40 7C 48 5C 54 74 7C 54 44 7C 10 1C 74 54 5C 74 54 7C
26B4 1C 24 44 7C 54 7C 7C 54 5C F8 A5 8E 18 69 01 85 8E D8 29 0F D0 0E A5 C3 38
26CD E9 06 85 C3 A5 C4 38 E9 10 85 C4 A5 06 AA BD E1 26 85 06 60 03 07 04 05 01
26E6 00 60 6C 66 64 00 00 62 6D 64 63 00 A0 3D 84 93 85 5C 60 20 91 27 4C E5 27
26FF 2A 00 65 5F 66 64 00 00 00 00 00 67 62 68 69 7D 00 00 70 70 6A 70 79 00 7D
2718 64 61 00 75 70 70 70 00 00 71 70 6A 71 79 00 7D 64 61 00 71 70 70 70 00 00
2731 72 70 6A 72 79 00 7D 64 61 00 70 75 70 70 00 00 73 70 6A 73 79 00 7D 64 61
274A 00 70 71 70 70 00 00 74 70 6A 74 79 00 7D 64 61 00 70 70 75 70 00 00 75 70
2763 6A 75 79 00 7D 64 61 00 70 70 70 75 00 00 00 00 00 00 00 64 6B 7D 64 00 70
277C 70 70 71 00 78 A9 E3 8D 14 03 A9 24 8D 15 03 A9 81 8D 28 03 58 A2 03 A9 00
2795 95 4A 95 5B 95 92 CA 10 F7 60 A5 A1 85 28 A5 A0 85 27 20 91 27 A9 55 85 94
27AE A9 FF 4C 2C 19 A9 01 85 93 85 94 60 A9 38 85 95 A2 08 60 A2 05 A9 FF 20 2C
27C7 19 CA 10 F8 A9 5C 85 5B 85 5E 60 A2 08 A9 00 95 A2 CA 10 FB 60 A2 03 A9 00
27E0 95 BA CA 10 FB A9 1B 85 5B A0 00 84 92 60 A9 42 85 5C A0 00 84 93 60 2A 2A
27F9 2A 2A 2A 2A 2A 2A 2A 2A 6B 67 11 9D 9D 9D 9D 9D 9D 00 1D 1D 05 00 00 05 6B
2812 68 00 6C 67 00 66 00 6A 00 69 69 69 69 69 69 69 69 00 6D 6D 6D 6D 6D 6D 6D
282B 6D 00 05 6B 67 1F 36 36 36 36 36 36 36 36 36 36 36 36 36 36 36 36 00 05 6B
2844 67 1F 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 00 05 76 77 78 79 7A
285D 00 7B 1F 7E 7F A0 7E 00 05 7B B0 B0 B0 B0 B0 7B 00 7B 7B B3 1C 7D 05 BA 7B
2876 1F BD A1 A2 A3 A4 00 00 AE AE AE AE AE AE AE AE AE AE AE AE AE AE 00 2D 2D
288F 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 00 13 00 1F 00 90 00 3A 3A
28A8 3A 00 3B 3B 3B 00 1C 00 6B 67 11 9D 9D 00 1C 38 00 2D 39 00 2B 2C 00 2D AF
28C1 A3 A4 BD BD 2D BC 2D A5 A2 2D BD A5 AC A3 A5 2D 00 12 9F 40 41 42 43 44 45
28DA 46 47 48 49 92 00 12 1E 4A 4B 4C 4D 4E 4F 50 51 52 53 92 00 12 1F 54 55 56
28F3 57 58 59 5A 5B 5C 5D 92 00 01 FE 24 DB A0 7E 87 AE 58 94 7E 86 8F 89 B6 87
290C 3C 86 B9 9A 51 9B 89 B6 7E 8A B9 96 BC 9C 89 B6 7E 8A B9 9E 8A D2 84 E0 89
2925 AE 9C 8A EE 7E 86 A2 8D 8D 94 8D 8D 8D A2 8D BF 8D 7E 87 AE 58 94 7E 86 8F
293E 7F A6 88 94 AA 9D 7E 87 00 90 9E 57 5E 65 6E 65 A0 03 03 0B 0B 17 24 24 14
2957 30 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 2D 42 10
```

Listing 2. »Penco« (Fortsetzung)

```
2970 1B 1B 19 00 A0 03 03 0B 0B 17 24 24 14 03 01 01 01 01 01 01 01 01 01 01 01
2989 01 01 01 01 01 01 01 01 01 01 01 01 10 1B 1B 19 00 A0 2D A2 09 B9 A8 94 09
29A2 08 99 A8 94 98 CA 10 F4 98 38 E9 08 A8 10 EB 60 A9 00 85 96 A9 1B 85 0F A9
29BB 10 85 0E 20 25 1B 29 07 AA BD D8 17 48 BD F8 17 20 DD 29 68 20 DD 29 E0 27
29D4 F0 04 F0 20 D0 E4 E6 96 60 48 4A 90 0A A5 0E C9 20 B0 0F E6 0E 90 0B 4A 90
29ED 08 A5 0E C9 02 90 02 C6 0E 68 29 F8 A8 20 2F 1D 4C 00 19 00 00 00 00 01 01
2A06 01 01 00 00 05 05 01 01 01 01 05 05 05 05 01 01 01 01 05 05 01 01 01 01 01
2A1F 01 01 01 01 01 01 01 01 01 00 00 00 00 00 00 00 00 00 00 10 12 00 00 00 00
2A38 10 12 18 21 00 00 00 00 09 13 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0
2A51 33 00 F2 E5 F0 3C 2A 2B 2D 2C 2A 2A 2D 2E 42 40 3E 3C 3A 01 FE 12 ED 1F 22
2A6A 28 25 10 10 11 11 37 46 C3 D2 02 02 28 28 04 20 04 20 A5 8E 29 0F 4A 85 06
2A83 0A 0A 0A A8 A2 07 B9 00 2A 95 2B B9 28 2A 95 37 C8 CA 10 F2 A9 00 85 A1 85
2A9C A0 A9 3C 85 33 60 A2 07 86 07 20 AF 2A A6 07 CA 10 F6 60 B4 2B F0 FB B9 4F
2AB5 2A 85 10 B9 56 2A 85 11 6C 10 00 20 62 1B A6 07 A9 06 95 83 95 2B A5 61 95
2ACE 42 A5 62 95 73 A9 45 95 7B A9 3A 20 09 2B A5 06 91 63 A0 00 91 63 60 B5 37
2AE7 C5 A1 B0 04 A9 01 95 2B 60 B5 7B F0 03 D6 7B 60 B5 42 85 61 B5 73 85 62 B4
2B00 83 89 30 11 D6 83 B9 5E 2A A0 00 91 61 18 69 01 A0 12 91 61 60 C8 B1 61 C9
2B19 2E D0 03 98 91 61 A9 02 95 2B A5 61 18 69 12 90 02 F6 73 95 42 60 EA EA EA
2B32 EA A5 0E 85 CE A5 0F 85 CF B5 7B 85 71 A9 05 85 8F B5 42 85 61 85 63 B5 73
2B4B 85 62 85 64 B5 83 85 70 30 0C A0 00 B1 61 F0 27 C9 36 B0 23 90 25 B5 7B 20
2B64 A3 19 A0 00 B1 63 C9 36 90 0B B1 61 C9 36 B0 0E 98 91 61 F0 09 B1 61 C9 36
2B7D 90 07 98 91 63 98 95 2B 50 B5 37 F0 03 D6 37 60 A5 70 10 18 29 7F 95 83 85
2B96 70 A9 00 A8 91 63 A5 70 91 61 A5 61 95 42 A5 62 95 73 60 20 25 1B C9 D0 B0
2BAF 44 C6 8F D0 01 60 A5 71 20 A3 19 A0 00 B1 61 A8 C9 1F B0 03 4C 57 2C C9 38
2BC8 B0 2A C9 36 90 22 20 25 1B C5 C3 90 1B C0 36 F0 05 A9 ED 20 A3 19 A5 61 85
2BE1 63 A5 62 85 64 20 99 22 A6 07 A9 04 95 37 60 C9 30 B0 FB A5 63 85 61 A5 64
2BFA 85 62 20 25 1B C5 C4 B0 19 20 25 1B 29 03 AA BC 64 2A 84 71 BD 68 2A 85 70
2C13 A6 07 95 83 94 7B 4C B0 2B 20 85 2C A5 D1 38 E5 CF 85 6F A5 D0 38 E5 CE 85
2C2C 6E 10 02 49 FF 85 60 A5 6F 10 02 49 FF 38 E5 60 90 0C A5 6F 10 04 A2 02 D0
2C45 0E A2 03 D0 0A A5 6E 10 04 A2 00 F0 02 A2 01 4C 09 2C 48 A5 61 85 67 A5 62
2C5E 18 69 94 85 68 A6 70 E8 8A A0 00 91 63 E8 8A 91 61 A5 06 91 67 A6 07 A5 70
2C77 09 80 95 83 68 F0 06 A8 68 68 4C 34 21 60 A5 62 38 E9 10 85 D0 A5 61 38 E9
2C90 23 B0 02 C6 D0 85 D1 46 D0 56 D1 A5 D1 [illegible] 00 C9 09 90 06 C8 38 E9 09 D0 F6
2CA9 0A 0A 85 D0 98 0A 85 D1 60 D6 37 F0 45 B5 37 4A B0 01 60 B5 42 85 61 B5 73
2CC2 85 62 A0 00 B1 61 C9 2B F0 07 C9 2C D0 23 A9 2B 2C A9 2C 91 61 B5 37 C9 08
2CDB B0 DE A5 62 09 84 85 62 B1 61 29 07 C5 06 D0 03 A9 06 2C A5 06 91 61 60 98
2CF4 95 2B A9 40 4C 99 24 A9 02 95 2F 60 A0 12 B1 29 A2 03 DD 68 2A F0 10 CA 10
2D0D F8 A5 27 85 CE A5 28 85 CF A6 07 4C 3B 2B A5 2A 09 84 85 2A A0 00 A5 06 91
2D26 29 A9 07 A6 07 95 2B A5 29 95 42 A5 2A 29 11 95 73 A9 FF 95 83 60 B5 37 F0
2D3F 03 D6 37 60 B5 42 85 61 B5 73 85 62 B4 83 C8 C0 06 90 03 4C DA 2E 98 95 83
2D58 C9 05 90 03 A9 20 2C A9 01 95 37 4C 05 2B A0 00 84 6F A2 07 B5 2B D0 01 C8
2D71 C9 05 D0 03 C8 E6 6F CA 10 F1 C0 07 90 1D A5 6F D0 08 C0 08 D0 04 E8 4C 27
2D8A 21 A0 02 A2 07 B5 2B C9 05 D0 09 A9 01 95 2B 88 D0 01 60 CA 10 EF A2 07 B5
2DA3 2B C9 02 F0 04 CA 10 F7 60 A9 03 95 2B 20 25 1B 29 03 A8 A2 03 B9 6C 2A 95
2DBC 27 C8 C8 C8 C8 CA 10 F4 60 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2DD5 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2DEE 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E07 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E20 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E39 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E52 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E6B 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E84 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2E9D 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2EB6 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A C6 33 D0 13 F8 A5 A1 18 69 01 85 A1 A5
2ECF A0 69 00 85 A0 D8 A9 3C 85 33 60 20 91 27 20 80 19 A2 07 BD E7 1A 85 61 BD
2EE8 EF 1A 85 62 BD D7 1A 85 63 BD DF 1A 85 64 A0 00 B1 61 F0 07 91 63 A9 10 20
2F01 2C 19 C8 C0 10 90 F0 CA 10 D7 F8 A5 A1 38 E9 08 85 08 A5 A0 E9 00 85 07 D8
2F1A A2 02 A9 01 9D 73 94 CA 10 FA A0 02 A2 08 A9 73 85 61 A9 10 85 62 20 C3 24
2F33 A5 07 D0 0D A5 08 29 F0 4A 4A 4A 4A AA E0 07 90 02 A2 06 8A 48 A9 91 85 61
2F4C A9 94 85 62 A5 61 18 69 24 85 61 90 02 E6 62 CA 10 F2 A0 0F A9 01 91 61 88
2F65 10 FB 20 C1 27 A5 61 18 69 0B 85 63 A5 62 69 00 49 84 85 64 68 AA BD C6 2F
2F7E 85 66 BD CD 2F 85 67 F8 A5 67 38 E9 01 85 67 A5 66 E9 00 85 66 D8 A0 03 A2
2F97 67 A5 63 85 61 A5 64 85 62 20 C3 24 A9 01 20 99 24 20 AC 24 A9 06 20 2C 19
2FB0 A5 67 05 66 D0 CF 68 68 68 68 20 91 27 A9 FF 20 2C 19 EA 4C 44 25 50 10 05
2FC9 01 00 00 00 00 00 00 00 50 25 01 A5 69 C9 2B B0 02 E6 6F 60 2A 2A 2A 2A 2A
2FE2 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A
2FFB 2A 2A 2A 2A 2A BA BA B4 32 5A 1A 1A 1A 08 00 04 C8 1B 4D 88 07 04 FA BA BA
```

Listing 2. »Penco« (Schluß)

Eingesperrt!

Out-Break ist ein Grafik-Adventure, bei dem Sie sich aus einem Haus ins Freie retten müssen. Doch das ist nicht so einfach, denn es müssen Bomben entschärft, Spinnen vernichtet und Abgründe überwunden werden.

Im Laufe des Spiels werden Sie feststellen, daß dies nicht die einzigen Hindernisse sind. In das Programm ist ein kleines Videospiel eingebaut und es gibt drei Schwierigkeitsstufen. Bei jedem neuen Start entstehen andere Variationen des Spiels. »OUT-BREAK« ist ein Spiel für den VC 20 mit 16 KByte RAM-Erweiterung.

Hier einige Hilfen zur Lösung des Spiels:

Spinnen können den Schaum von Feuerlöschern nicht vertragen, man sollte es hier mit den Tasten B, N und SHIFT probieren. In blauen Räumen verliert man leicht die Orientierung. Verschlossene Türen lassen sich nur mit einem Schlüssel öffnen. Schier unüberwindliche Abgründe lassen sich mit einer Leiter überwinden. Manchmal findet man auf weggeworfenen Zetteln brauchbare Informationen.

Um im Dunkeln zu sehen, muß man eine Lampe anmachen. Bei Schwierigkeitsgrad 2 und 3 kann die Tür in die Freiheit nur mit einem Spezialschlüssel aus einem Tresor geöffnet werden. Die Freiheit lockt hinter der grünen Tür.

Das Programm besteht aus zwei Teilen. Der erste Teil (Listing 1) ist ein Basic-Lader, der die Daten für den undefinierten Zeichensatz in den Speicher schreibt. Dieser Teil muß immer als erstes geladen und gestartet werden. Vorsicht! Durch den NEW-Befehl in Zeile 140 löscht sich dieser Teil von selbst, nachdem er seine Aufgabe verrichtet hat. Jetzt muß der zweite Teil (Listing 2), das eigentliche Programm, geladen und gestartet werden. Es versteht sich wohl von selbst, daß beide Teile vor dem Ausprobieren sicherheitshalber erst auf Kassette oder Disk gespeichert werden sollten.

(Gudrun Disser/ev)

Programmablaufplan:

Zeilen		Funktion
160	=	Rücksetzen aller Variablen und Umschalten in Normalmodus
170 - 220	=	Festlegen der Konstanten
230 - 300	=	Anfangsbild
310 - 410	=	Auswahl des Schwierigkeitsgrades und Starten der Zeit
420 - 840	=	Festlegen der Stringvariablen
900 - 1010	=	Starten der Stoppuhr, wenn Tresor oder Bombe gefunden wurden.
1020 - 1380	=	Abfrage und Auswertung, welcher Befehl eingegeben wurde
1390 - 2420	=	Auswahl und Definition der Räume
2430 - 3000	=	Anweisung bei falscher Eingabe oder Spielhinweise
3010 - 3070	=	Anzeigen der Stoppuhr
3080 - 3230	=	Schlußbild
3240 - 3740	=	Spinnenspiel
3750 - 4040	=	Spielende und Erklärung, warum verloren wurde
4050 - 4480	=	Definition, welcher Gegenstand sich in welchem Raum befindet.

```
100 REM *** OUT-BREAK TEIL 1 ***                  <026>
110 PRINT"{CLR,9DOWN,4SPACE}COPYRIGHT BY{8
    SPACE,DOWN,2SPACE}GUDRUN DISSER"              <044>
120 FOR I=7168 TO 7168+391:READ A:POKE I,A
    :NEXT                                         <032>
130 PRINT"{CLR,RED,5DOWN,4SPACE}LADEN SIE
    JETZT{3SPACE,DOWN,6SPACE}OUT-BREAK 2"         <254>
140 POKE 44,30:POKE 7680,0:NEW                    <088>
150 DATA 0, 0, 0, 0, 0, 0, 0, 0                   <068>
160 DATA 1, 1, 3, 7, 15, 15, 63, 63               <226>
170 DATA 0, 192, 192, 224, 240, 240, 252,
    252                                           <170>
180 DATA 63, 63, 63, 63, 31, 15, 7, 7             <199>
190 DATA 252, 252, 252, 252, 248, 240, 224
    , 224                                         <053>
200 DATA 3, 4, 8, 48, 64, 128, 128, 128           <137>
210 DATA 0, 96, 156, 14, 252, 12, 30, 63          <064>
220 DATA 51, 55, 51, 55, 55, 63, 63, 55           <114>
230 DATA 55, 55, 55, 51, 63, 63, 63, 33           <197>
240 DATA 0, 0, 0, 0, 0, 0, 0, 12                  <104>
250 DATA 0, 0, 0, 0, 0, 48, 112, 240              <014>
260 DATA 255, 139, 219, 217, 255, 0, 0, 0         <062>
270 DATA 240, 240, 240, 240, 240, 112, 48,
    0                                             <025>
280 DATA 255, 255, 255, 255, 255, 126, 48,
    0                                             <082>
290 DATA 255, 255, 243, 161, 0, 0, 0, 0           <076>
300 DATA 255, 255, 255, 255, 98, 0, 0, 0          <113>
310 DATA 3, 3, 3, 15, 31, 23, 103, 157            <144>
320 DATA 192, 192, 192, 240, 248, 252, 215
    , 77                                          <219>
330 DATA 3, 3, 3, 3, 3, 3, 3, 3                   <248>
340 DATA 192, 192, 192, 192, 192, 192, 192
    , 192                                         <252>
350 DATA 101, 63, 53, 85, 93, 43, 51, 17          <097>
360 DATA 4, 15, 18, 28, 63, 55, 77, 95            <116>
370 DATA 240, 40, 124, 204, 214, 250, 86,
    124                                           <060>
380 DATA 164, 246, 188, 208, 152, 232, 144
    , 144                                         <023>
390 DATA 165, 218, 151, 109, 89, 146, 75,
    181                                           <116>
400 DATA 255, 254, 252, 252, 248, 200, 192
    , 128                                         <024>
410 DATA 255, 127, 127, 63, 63, 23, 3, 1          <020>
420 DATA 1, 2, 2, 12, 48, 64, 128, 128            <022>
430 DATA 255, 199, 215, 199, 239, 231, 239
    , 231                                         <134>
440 DATA 0, 0, 0, 124, 131, 0, 0, 0               <252>
450 DATA 0, 0, 0, 0, 195, 60, 0, 0                <014>
460 DATA 0, 0, 0, 0, 255, 0, 0, 0                 <238>
470 DATA 0, 0, 0, 0, 0, 0, 0, 0                   <134>
480 DATA 255, 255, 255, 255, 255, 255, 255
    , 255                                         <052>
490 DATA 0, 0, 0, 0, 0, 0, 0, 0                   <154>
500 DATA 16, 40, 40, 16, 16, 24, 16, 24           <231>
510 DATA 129, 66, 60, 165, 126, 24, 36, 19
    5                                             <064>
520 DATA 1, 1, 31, 17, 1, 1, 1, 1                 <135>
530 DATA 64, 64, 64, 224, 16, 236, 60, 0          <222>
540 DATA 16, 16, 16, 16, 56, 124, 254, 0          <104>
550 DATA 16, 16, 16, 16, 16, 16, 16, 0            <017>
560 DATA 255, 136, 218, 216, 217, 218, 255
    , 255                                         <038>
570 DATA 255, 145, 183, 145, 189, 145, 255
    , 255                                         <155>
580 DATA 255, 17, 85, 81, 83, 21, 255, 255 <180>
590 DATA 255, 247, 213, 227, 193, 227, 213
    , 247                                         <164>
600 DATA 127, 85, 106, 53, 42, 53, 26, 21 <060>
610 DATA 254, 170, 86, 172, 84, 172, 88, 1
    68                                            <245>
620 DATA 26, 13, 10, 13, 10, 13, 10, 15           <049>
630 DATA 88, 176, 80, 176, 80, 176, 80, 24
    0                                             <114>
© 64'er
```

Listing 1. »Out-Break« Ladeprogramm für Grafikzeichen

```
10 REM  ***** OUT-BREAK 2 ****                      <192>
20 REM                                              <082>
30 REM                                              <092>
100 REM ***********************                     <157>
110 REM *     COPYRIGHT BY     *                    <021>
120 REM *    GUDRUN  DISSER    *                    <238>
130 REM *     SCHULSTR. 42     *                    <072>
140 REM *   6451 MAINHAUSEN    *                    <130>
150 REM ***********************                     <207>
160 CLR:POKE 36869,192                              <037>
170 C=INT(RND(1)*4)+1                               <183>
180 V=INT(RND(1)*899)+100                           <164>
190 V2$=STR$(V):V2$=RIGHT$(V2$,3)                   <045>
200 V3$=LEFT$(V2$,1):V4$=MID$(V2$,2,1):V5$
    =RIGHT$(V2$,1)                                  <201>
210 F=33792:T=36878:T1=36874:T2=36875:T3=3
    6876:T4=36877                                   <165>
220 FF=INT(RND(1)*7)+24                             <017>
230 POKE 36879,25:Z$="{SPACE,RVSON,20SPACE
    ,RVOFF}"                                        <138>
240 PRINT"{CLR,WHITE}"Z$:PRINT"{RVSON,7SPA
    CE}OUT-BREAK{6SPACE,RVOFF}";:PRINT Z$           <012>
250 PRINT"{3DOWN,RVOFF,PURPLE,2SPACE}SCHWI
    ERIGKEITSGRAD"                                  <147>
260 PRINT"{2DOWN,2SPACE,RVSON,BLUE,SPACE}1
    {SPACE,RVOFF,SPACE,GREEN}EINFACH"               <164>
270 PRINT"{DOWN,2SPACE,RVSON,BLUE,SPACE}2{
    SPACE,RVOFF,SPACE,GREEN}SCHWER"                 <249>
280 PRINT"{DOWN,2SPACE,RVSON,BLUE,SPACE}3{
    SPACE,RVOFF,SPACE,GREEN}SEHR SCHWER             <105>
290 PRINT"{4DOWN,WHITE}"Z$:PRINT"{RVSON,4S
    PACE}BITTE WAEHLEN{5SPACE,RVOFF}";:PRI
    NT Z$                                           <103>
300 L6=4557:FOR H=4096 TO 4160:POKE H+F,2:
    POKE L6+F,2:L6=L6-1:GOSUB 3070:NEXT             <227>
310 GET G                                           <248>
320 IF G=1 THEN X=20:GOTO 370                       <154>
330 IF G=2 THEN X=10:GOTO 370                       <194>
340 IF G=3 THEN X=5:U=2:X2=4:GOTO 370               <088>
350 L7=L7+1:IF L7=300 THEN L7=0:PRINT"{CLR
    }":FOR H=1 TO 3000:NEXT:GOTO 160                <147>
360 GOTO 310                                        <066>
370 GOTO 390                                        <204>
380 L6=4557:FOR H=4096 TO 4160:POKE H,32:P
    OKE L6,32:L6=L6-1:GOSUB 3070:NEXT               <218>
390 PRINT"{CLR}":A=2:TY=1:TI$="000000"              <086>
400 PRINT"{CLR,BLACK}"                              <220>
410 POKE 36869,207                                  <047>
420 A1$="{HOME,RVSON}M{DOWN}M{DOWN}M{DOWN}
    M{DOWN}M"                                       <100>
430 A2$="{HOME,21RIGHT,RVSON}N{DOWN,2LEFT}
    N{DOWN,2LEFT}N{DOWN,2LEFT}N{DOWN,2LEFT
    }N"                                             <053>
440 B1$="{HOME,5DOWN,5RIGHT,RVSON}OTTTTTTT
    TTTP"                                           <137>
450 B2$="{HOME,15DOWN,5RIGHT,RVSON}LGGGGGG
    GGGGG"                                          <236>
460 B3$="{HOME,6DOWN,5RIGHT,RVSON}G{DOWN,L
    EFT}G{DOWN,LEFT}G{DOWN,LEFT}G{DOWN,LEF
    T}G{DOWN,LEFT}G{DOWN,LEFT}G{DOWN,LEFT}
    G{DOWN,LEFT}G"                                  <014>
470 B4$="{HOME,6DOWN,16RIGHT,RVSON}H{DOWN,
    LEFT}H{DOWN,LEFT}H{DOWN,LEFT}H{DOWN,LE
    FT}H{DOWN,LEFT}H{DOWN,LEFT}H{DOWN,LEFT
    }H{DOWN,LEFT}H"                                 <194>
480 A3$="{HOME,16DOWN,4RIGHT,RVSON}N{DOWN,
    2LEFT}N{DOWN,2LEFT}N{DOWN,2LEFT}N{DOWN
    ,2LEFT}N"                                       <182>
490 A4$="{HOME,16DOWN,17RIGHT,RVSON}M{DOWN
    }M{DOWN}M{DOWN}M"                               <095>
500 I$="{HOME,10DOWN,8RIGHT,RVSON}OTP{DOWN
    ,3LEFT}G{RIGHT}H{DOWN,3LEFT}G{RIGHT}H{
    DOWN,3LEFT}G{RIGHT,RVOFF}%{RVSON,DOWN,
    3LEFT}G{RIGHT}H{DOWN,3LEFT}LGG"                 <160>
510 J1$="{HOME,6DOWN,RVSON}HM{DOWN,2LEFT}H
    {RIGHT}M{DOWN,3LEFT}H{2RIGHT}M{DOWN,4L
    EFT}HTTP{DOWN,4LEFT}H{2RIGHT}H{DOWN,4L
    EFT}H{2RIGHT}H{DOWN,4LEFT}H{2RIGHT}H{D
    OWN,4LEFT}H{2RIGHT}H"                           <145>
520 J2$="{HOME,13DOWN,RVSON}H{2RIGHT}H{DOW
    N,4LEFT}H{2RIGHT}H{DOWN,4LEFT}H{2RIGHT
    }H{DOWN,4LEFT}H{2RIGHT}H"                       <103>
530 J3$="{HOME,17DOWN,RVSON}HTTT{UP}N{DOWN
    ,LEFT,DOWN,4LEFT}H{DOWN,LEFT}H{DOWN,LE
    FT}N"                                           <212>
540 K1$="{HOME,6DOWN,20RIGHT,RVSON}NG{DOWN
    ,3LEFT}N{RIGHT}G{DOWN,4LEFT}N{2RIGHT}G
    {DOWN,4LEFT}OTTG{DOWN,4LEFT}G{2RIGHT}G
    {DOWN,4LEFT}G{2RIGHT}G"                         <229>
550 K2$="{HOME,12DOWN,RVSON,18RIGHT,RVSON}
    G{2RIGHT}G{DOWN,4LEFT}G{2RIGHT}G{DOWN,
    4LEFT}G{2RIGHT}G{DOWN,4LEFT}G{2RIGHT}G
    {DOWN,5LEFT}MLGGG"                              <157>
560 K3$="{HOME,16DOWN,RVSON,21RIGHT}G{DOWN
    ,LEFT}G{DOWN,LEFT}G{DOWN,LEFT}G{DOWN,L
    EFT}M"                                          <005>
570 C1$="{HOME,5DOWN,5RIGHT,RVSON}G{DOWN,L
    EFT}OPM{DOWN,3LEFT}GH{RIGHT}M{DOWN,4LE
    FT}GH{RIGHT}H{DOWN,4LEFT}GH{RIGHT}H{DO
    WN,4LEFT}GH{RIGHT}H{DOWN,4LEFT}GH{RIGH
    T}H{DOWN,4LEFT}GH{RIGHT}H"                      <092>
580 C2$="{HOME,12DOWN,5RIGHT,RVSON}GH{RIGH
    T}H{DOWN,4LEFT}GH{RIGHT}N{DOWN,4LEFT}L
    GN{DOWN,3LEFT}G"                                <200>
590 C3$="{HOME,5DOWN,16RIGHT,RVSON}H{DOWN,
    3LEFT}NOP{DOWN,4LEFT}N{RIGHT}GH{DOWN,4
    LEFT}G{RIGHT}GH{DOWN,4LEFT}G{RIGHT}GH{
    DOWN,4LEFT}G{RIGHT}GH"                          <243>
600 C4$="{HOME,11DOWN,13RIGHT,RVSON}G{RIGH
    T}GH{DOWN,4LEFT}G{RIGHT}GH{DOWN,4LEFT}
    M{RIGHT}GH{DOWN,3LEFT}MLG{DOWN,LEFT}H"          <104>
610 D1$="{HOME,9DOWN,8RIGHT,RVSON}GGGGGG{D
    OWN,7LEFT}HM{4RIGHT}NG{DOWN,8LEFT}H{RI
    GHT}M{2RIGHT}N{RIGHT}G{DOWN,8LEFT}H{2R
    IGHT}OP{2RIGHT}G"                               <164>
620 D7$="{HOME,9DOWN,8RIGHT,RVSON}GGGGGG{D
    OWN,7LEFT}HM{4RIGHT}NG{DOWN,8LEFT}H{RI
    GHT,SPACE,2RIGHT,SPACE,RIGHT}G{DOWN,8L
    EFT}H{4SPACE,2RIGHT}G"                          <227>
630 D2$="{HOME,13DOWN,7RIGHT,RVSON}H{2RIGH
    T}LG{2RIGHT}G{DOWN,8LEFT}H{RIGHT}N{2RI
    GHT}M{RIGHT}G{DOWN,8LEFT}GN{4SPACE}ML"          <043>
640 E$="{HOME,8DOWN,9RIGHT,RVSON}TTTT{4LEF
    T,4DOWN}GGGG"                                   <046>
650 F$="{HOME,13DOWN,9RIGHT,RVOFF,BLUE}I{R
    ED}J{DOWN,2LEFT,BLUE}K{RED}L{BLACK}"            <123>
660 G1$="{HOME,18DOWN,3RIGHT,RVSON,PURPLE}
    NOTTTTTTTTTTTTPM{DOWN,17LEFT}N{RIGHT}G
    {12RIGHT}H{RIGHT}M"                             <094>
670 G2$="{HOME,20DOWN,2RIGHT,RVSON,PURPLE}
    TTTTTTTTTTTTTTTTTT{BLACK}"                      <111>
680 H1$="{HOME,10DOWN,12RIGHT,RVSON,PURPLE
    }HCG{DOWN,3LEFT}HCG{DOWN,3LEFT}HCG{DOW
    N,3LEFT}HCG{DOWN,3LEFT}HCG{DOWN,3LEFT}
    HCG{DOWN,3LEFT}HCG{BLACK}"                      <210>
690 H2$="{HOME,17DOWN,9RIGHT,RVSON,PURPLE}
    HCG{DOWN,3LEFT}PCO{DOWN,3LEFT}HCG{DOWN
    ,3LEFT}PCO{DOWN,3LEFT}HCG{BLACK}"               <241>
700 M$="{HOME,15DOWN,11RIGHT,RVOFF,GREEN}E
    {BLACK,DOWN,2LEFT}AB{DOWN,2LEFT}CD"             <110>
710 L$="{HOME,12DOWN,11RIGHT,RVOFF,RED}F{D
    OWN,LEFT}G{DOWN,LEFT}H{BLACK}"                  <174>
720 O$="{GREEN}UXV{DOWN,3LEFT}TXXV{DOWN,4L
    EFT}XXXV{DOWN,4LEFT}TXXW{DOWN,3LEFT,BL
    ACK}RS{DOWN,2LEFT}RS{DOWN,2LEFT}PQ"             <087>
730 N$="{HOME,10DOWN,RIGHT,RVSON,BLUE}BRAV
    O!!!"                                           <165>
740 N1$="{RVSON,BLUE,SPACE}DU BIST FREI"            <041>
750 P$="{RVSON,RED,SPACE}OHNE SCHLUESSEL G
    EHT{DOWN,7SPACE}DAS NICHT"                      <194>
760 Q$="{RVSON,BLUE}DAS GEHT NICHT!!!{BLAC
    K}"                                             <068>
770 R$="{RVSON,YELLOW}IM DUNKELN FINDEST D
    U{SPACE,DOWN,2SPACE}DICH NICHT ZURECHT
    "                                               <070>
780 S$="{YELLOW}DU KONNTEST DIE BOMBE{SPAC
    E,DOWN,2SPACE}NICHT RECHTZEITIG{3SPACE
    ,DOWN,4SPACE}BESEITIGEN!!!!"                    <116>
790 T$="{YELLOW}DU BIST IN DEN SCHACHT{DOW
    N}GEFALLEN UND HAST DAS{SPACE,DOWN,2SP
    ACE}SPIEL VERLOREN!!!!"                         <189>
800 U$="{RVSON,RED,6SPACE}SEHR GUT!!{6SPAC
    E,2DOWN,BLUE,2SPACE}DU HAST DIE BOMBE{
    2SPACE,DOWN,7SPACE}BESEITIGT.{BLACK}"           <162>
810 M1$="{RVSON,PURPLE,2SPACE}DAS GEHT NIC
    HT DU{3SPACE,DOWN,SPACE}MUSST ERST DIE
    BOMBE{SPACE,DOWN,5SPACE}LOSWERDEN!!"            <208>
820 V$="{HOME,13DOWN,12RIGHT,BLUE})*+{DOWN
    ,3LEFT}!,!{DOWN,3LEFT}!!!{RVOFF,BLACK}
    "                                               <119>
```

```
830 V1$="{HOME,13DOWN,12RIGHT,BLACK}!!!{DO
    WN,3LEFT}!£!{DOWN,3LEFT}!!!{BLUE,4LEFT
    ,2UP,RVSON}Γ{DOWN,LEFT}Γ{DOWN,LEFT}Γ{R
    VOFF,RLACK}"                              <027>
840 W$="{HOME,18DOWN,16RIGHT,PURPLE}-.{DOW
    N,2LEFT}/0"                               <032>
850 GOSUB 4050                                <228>
860 B=0:C1=C*A:B$="":A$="":D2=0               <158>
870 IF A=15 AND M=0 THEN GOSUB 2660:GOSUB
    4050                                      <190>
880 IF D1=1 THEN PRINT"{UP,WHITE}"            <187>
890 PRINT"{HOME,22DOWN,RVSON}"CHR$(63);CHR
    $(32);CHR$(122);                          <082>
900 IF A<>13 THEN T5=0:X2=4                   <099>
910 IF M1>0 OR M=1 THEN 930                   <145>
920 IF A=15 THEN M1=1:M2=VAL(TI$):M=3         <221>
930 IF M1=1 THEN 3010                         <160>
940 IF T5=1 THEN 960                          <175>
950 IF A=13 AND G=3 THEN T5=1:M4=VAL(TI$)     <149>
960 IF T5=1 THEN 3030                         <062>
970 GET Y$                                    <098>
980 IF T5=1 AND X2=<0 THEN A=INT(RND(1)*17
    )+1:GOSUB 4050:GOTO 860                   <187>
990 IF M1=1 AND X=<0 THEN Z=3:GOTO 3750       <073>
1000 IF M3=>(M4+5)AND T5=1 THEN POKE 4106+
     X2,32:X2=X2-1:M4=M3                      <133>
1010 M3=VAL(TI$):IF M3=>(M2+5)AND M1=1 THE
     N POKE 4096+X,32:X=X-1:M2=M3             <111>
1020 IF Y$=""THEN 970                         <161>
1030 PRINT CHR$(157)Y$CHR$(122);              <215>
1040 POKE T,1:POKE T3,200                     <093>
1050 IF Y$=CHR$(20)AND LEN(B$)=0 THEN 950     <029>
1060 IF Y$=CHR$(20)THEN B$=LEFT$(B$,LEN(B$
     )-1)                                     <249>
1070 B$=B$+Y$                                 <210>
1080 IF Y$=CHR$(20)THEN B$=LEFT$(B$,LEN(B$
     )-1)                                     <013>
1090 POKE T,0:POKE T3,0                       <086>
1100 IF Y$=CHR$(13)THEN 1120                  <062>
1110 GOTO 970                                 <198>
1120 PRINT"{CLR}"                             <092>
1130 A$=MID$(B$,6,4):B$=LEFT$(B$,4)           <246>
1140 IF B$="FARB"THEN GOSUB 3050:GOTO 1370    <052>
1150 IF B$="LAMP"AND MID$(A$,2,2)="AN"AND
     D=1 THEN D2=1:GOTO 1370                  <198>
1160 IF D1=1 AND B$<>"ZURU"THEN GOSUB 2530    <140>
1170 IF D1=1 THEN GOSUB 1210:GOTO 1370        <087>
1180 IF B$="RECH"THEN B=-1:GOTO 1370          <230>
1190 IF B$="LINK"THEN B=1:GOTO 1370           <128>
1200 IF LEFT$(B$,3)="VOR"THEN B=2:GOTO 137
     0                                        <062>
1210 IF B$="ZURU"THEN B=-2:GOTO 1370          <107>
1220 IF D1=1 THEN RETURN                      <050>
1230 IF B$="NIMM"AND A$="LEIT"AND A=5 THEN
     L=1:GOTO 1370                            <234>
1240 IF B$="LEGE"AND A$="LEIT"AND L=1 AND
     A=6 THEN L=3:GOTO 1370                   <028>
1250 IF B$="NIMM"AND A$="LAMP"AND(C1=3 OR
     C1=22 OR C1=42 OR C1=56)THEN D=1:GOTO
     1370                                     <008>
1260 IF B$="NIMM"AND A$="SCHL"AND A=6 AND
     L=3 THEN S5=1:GOTO 1370                  <183>
1270 IF S5=1 AND L<>3 THEN X=1:GOTO 3750      <152>
1280 IF B$="NIMM"AND A$="FEUE"AND(C1=11 OR
     C1=44 OR C1=6 OR C1=9)THEN F1=1:GOTO
     1370                                     <046>
1290 IF B$="NIMM"AND A$="BOMB"AND A=15 THE
     N M=1:GOTO 1370                          <179>
1300 IF B$="WIRF"AND A$="BOMB"AND A=6 AND
     M=1 THEN M=2:M1=2:GOSUB 2620:GOTO 137
     0                                        <163>
1310 IF A=13 AND LEFT$(B$,3)=V2$THEN T6=1:
     GOTO 1370                                <090>
1320 IF B$="NIMM"AND A$="SCHL"AND A=13 AND
     T6=1 THEN T6=0:S6=1:GOTO 1370            <087>
1330 IF A=10 AND B$="SIEH"AND A$="NACH"THE
     N:Z=2:GOTO 2870                          <195>
1340 IF A=10 AND B$="NIMM"AND A$="ZETT"AND
     Z=2 THEN Z=1:GOTO 1370                   <111>
1350 IF Z=1 AND B$="LIES"AND A$="ZETT"THEN
     GOTO 2890                                <145>
1360 GOSUB 2430                               <214>
1370 PRINT"{BLACK}":D1=0                      <046>
1380 IF A<>13 THEN T5=0:X2=4                  <071>
1390 ON A GOTO 1400,1460,1520,1570,1630,16
     90,1760,1830,1890,1940,1990,2040,2110
     ,2170,2230,2290,2350                     <087>
1400 IF B=1 THEN A=14                         <128>
1410 IF B=-1 THEN A=2                         <249>
1420 IF B=2 THEN A=6                          <238>
1430 IF B=-2 THEN GOSUB 2430                  <060>
1440 GOSUB 4050                               <054>
1450 GOTO 860                                 <004>
1460 IF B=1 THEN A=1                          <236>
1470 IF B=-1 THEN A=3                         <057>
1480 IF B=2 THEN A=8                          <046>
1490 IF B=-2 THEN GOSUB 2430                  <120>
1500 GOSUB 4050                               <114>
1510 GOTO 860                                 <064>
1520 IF B=1 THEN A=2                          <042>
1530 IF B=-1 THEN A=4                         <121>
1540 IF B=-2 OR B=2 THEN GOSUB 2430           <116>
1550 GOSUB 4050                               <166>
1560 GOTO 860                                 <116>
1570 IF B=1 THEN A=3                          <096>
1580 IF B=-1 THEN A=16                        <082>
1590 IF B=2 THEN A=5                          <152>
1600 IF B=-2 THEN GOSUB 2430                  <232>
1610 GOSUB 4050                               <226>
1620 GOTO 860                                 <176>
1630 IF B=2 AND S5=1 THEN A=9                 <002>
1640 IF B=2 AND S5=0 THEN GOSUB 2480          <199>
1650 IF B=-2 THEN A=4                         <051>
1660 IF B=1 OR B=-1 THEN GOSUB 2430           <100>
1670 GOSUB 4050                               <030>
1680 GOTO 860                                 <236>
1690 IF B=2 AND L=3 AND S5=1 THEN A=7         <034>
1700 IF B=2 AND L<>3 THEN Z=1:GOTO 3750       <108>
1710 IF B=-2 THEN A=1                         <099>
1720 IF B=2 AND S5=0 THEN GOSUB 2480          <023>
1730 IF B=1 OR B=-1 THEN GOSUB 2430           <170>
1740 GOSUB 4050                               <100>
1750 GOTO 860                                 <050>
1760 IF B=-1 THEN A=8                         <113>
1770 IF B=2 AND S5=1 THEN A=13:GOSUB 2780     <132>
1780 IF B=2 AND S5=0 THEN GOSUB 2480          <083>
1790 IF B=-2 THEN A=6                         <199>
1800 IF B=1 THEN GOSUB 2430                   <061>
1810 GOSUB 4050                               <172>
1820 GOTO 860                                 <122>
1830 IF B=1 THEN A=7                          <110>
1840 IF B=-1 THEN A=9                         <199>
1850 IF B=2 THEN A=12                         <094>
1860 IF B=-2 THEN A=2                         <255>
1870 GOSUB 4050                               <232>
1880 GOTO 860                                 <182>
1890 IF B=1 THEN A=8                          <172>
1900 IF B=-2 THEN A=5                         <051>
1910 IF B=-1 OR0B=2 THEN GOSUB 2430           <168>
1920 GOSUB 4050                               <026>
1930 GOTO 860                                 <232>
1940 IF B=1 THEN A=11                         <148>
1950 IF B=2 THEN A=16                         <210>
1960 IF B=-1 OR B=-2 THEN GOSUB 2430          <132>
1970 GOSUB 4050                               <076>
1980 GOTO 860                                 <026>
1990 IF B=1 THEN A=12                         <202>
2000 IF B=-1 THEN A=10                        <200>
2010 IF B=2 OR B=-2 THEN GOSUB 2430           <232>
2020 GOSUB 4050                               <126>
2030 GOTO 860                                 <076>
2040 IF B=-1 THEN A=11                        <248>
2050 IF B=2 AND S5=1 THEN A=15                <066>
2060 IF B=2 AND S5=0 THEN GOSUB 2480          <111>
2070 IF B=-2 THEN A=8                         <235>
2080 IF B=1 THEN GOSUB 2430                   <154>
2090 GOSUB 4050                               <198>
2100 GOTO 860                                 <148>
2110 IF B=-2 THEN A=7                         <015>
2120 IF B=2 AND S5=0 THEN GOSUB 2480          <171>
2130 IF B=2 AND S5=1 THEN A=14:TY=2           <017>
2140 IF B=-1 OR B=1 THEN GOSUB 2430           <140>
2150 GOSUB 4050                               <002>
2160 GOTO 860                                 <208>
2170 IF B=1 THEN A=1                          <184>
2180 IF B=-1 THEN A=15                        <166>
2190 IF B=-2 THEN A=13:GOSUB 2780             <243>
2200 IF B=2 THEN GOSUB 2430                   <239>
2210 GOSUB 4050                               <062>
2220 GOTO 860                                 <012>
```

Listing 2. Hauptprogramm für »Out-Break«.
Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
2230 IF B=2 THEN A=17                           <240>
2240 IF B=-2 THEN A=12                          <010>
2250 IF B=1 THEN A=14                           <216>
2260 IF B=-1 THEN GOSUB 2430                    <064>
2270 GOSUB 4050                                 <122>
2280 GOTO 860                                   <072>
2290 IF B=-1 THEN A=4                           <119>
2300 IF B=1 THEN A=17                           <022>
2310 IF B=-2 THEN A=10                          <066>
2320 IF B=2 THEN GOSUB 2430                     <105>
2330 GOSUB 4050                                 <184>
2340 GOTO 860                                   <134>
2350 IF B=-1 THEN A=16                          <090>
2360 IF B=-2 THEN A=15                          <156>
2370 IF B=2 AND(M=1 OR M=3)THEN GOSUB 2580 <069>
2375 IF B=2 AND G=1 AND S5=0 THEN GOSUB 24
     80:GOTO 2400                               <039>
2376 IF B=2 AND G=1 AND S5=1 AND(M=0 OR M=
     2)THEN 3080                                <165>
2380 IF B=2 AND S6=0 AND(M=0 OR M=2)THEN G
     OSUB 2830                                  <111>
2390 IF B=2 AND S6=1 AND(M=0 OR M=2)THEN 3
     080                                        <031>
2400 IF B=1 THEN GOSUB 2430                     <153>
2410 GOSUB 4050                                 <008>
2420 GOTO 860                                   <214>
2430 PRINT"{CLR,9DOWN,3RIGHT}";                 <130>
2440 FOR Q=1 TO LEN(Q$):PRINT"{RVSON}"MID$
     (Q$,Q,1);                                  <016>
2450 IF MID$(Q$,Q,1)=CHR$(32)THEN NEXT Q       <186>
2460 GOSUB 3230:NEXT Q                          <083>
2470 FOR H=1 TO 1500:NEXT:RETURN                <074>
2480 PRINT"{CLR,9DOWN}"                         <026>
2490 FOR Q=1 TO LEN(P$):PRINT"{RVSON}"MID$
     (P$,Q,1);                                  <056>
2500 IF MID$(P$,Q,1)=CHR$(32)THEN NEXT Q       <204>
2510 GOSUB 3230:NEXT Q                          <133>
2520 FOR H=1 TO 2000:NEXT:PRINT"{BLACK}":R
     ETURN                                      <252>
2530 PRINT"{CLR,9DOWN}"                         <076>
2540 FOR Q=1 TO LEN(R$):PRINT"{RVSON}"MID$
     (R$,Q,1);                                  <126>
2550 IF MID$(R$,Q,1)=CHR$(32)THEN NEXT Q       <062>
2560 GOSUB 3230:NEXT Q                          <185>
2570 FOR H=1 TO 2000:NEXT:PRINT"{WHITE}":R
     ETURN                                      <165>
2580 PRINT"{CLR,8DOWN}"                         <026>
2590 FOR Q=1 TO LEN(M1$):PRINT"{RVSON}"MID
     $(M1$,Q,1);                                <088>
2600 IF MID$(M1$,Q,1)=CHR$(32)THEN NEXT Q      <007>
2610 GOSUB 3230:NEXT Q:FOR H=1 TO 1000:NEX
     T:RETURN                                   <241>
2620 PRINT"{CLR,8DOWN}                          <254>
2630 FOR Q=1 TO LEN(U$):PRINT"{RVSON}"MID$
     (U$,Q,1);                                  <248>
2640 IF MID$(U$,Q,1)=CHR$(32)THEN NEXT Q       <250>
2650 GOSUB 3230:NEXT Q:FOR H=1 TO 1000:NEX
     T:RETURN                                   <025>
2660 FOR H=1 TO 2000:NEXT                       <156>
2670 IF G=1 THEN X1=100                         <046>
2680 IF G=2 THEN X1=50                          <101>
2690 IF G=3 THEN X1=25                          <171>
2700 PRINT"{CLR,5DOWN,RVSON,WHITE,2SPACE}V
     OR DIR LIEGT EINE{2SPACE,DOWN,2SPACE}
     BOMBE. DU HAST CA.{2SPACE}"                <215>
2710 IF G=1 THEN X1=100                         <086>
2720 PRINT"{RVSON}"X1;:PRINT"SEKUNDEN ZEIT
      UM{SPACE,DOWN,SPACE}SIE IN EINEN SCH
     ACHT{SPACE,DOWN,2SPACE}ZU WERFEN. NUR
      SO{DOWN,SPACE}"                           <184>
2730 PRINT"{RVSON,4SPACE}KANNST DU SIE{5SP
     ACE,DOWN,5SPACE}BESEITIGEN!!{BLACK,RV
     OFF}"                                      <182>
2740 FOR I=1 TO 100                             <186>
2750 POKE 36879,10:GOSUB 2770:POKE 36879,4
     0:GOSUB 2770:NEXT I                        <082>
2760 RETURN                                     <022>
2770 POKE T,6:POKE T3,250:FOR H=1 TO 10:NE
     XT:POKE T,0:POKE T3,0:RETURN               <095>
2780 IF G<>3 THEN RETURN                        <246>
2790 PRINT"{CLR,4DOWN,RVSON,2SPACE}DU HAST
      DURCH DAS{3SPACE,DOWN,SPACE}BETRETEN
      DES RAUMES{2SPACE}"                       <152>
2800 PRINT"{RVSON,SPACE}DIE ALARMANLAGE AU
     S-{SPACE,DOWN,SPACE}GELOEST. DEINE VE
     R-{2SPACE,DOWN,SPACE}FOLGER SIND DIR
     AUF{2SPACE}"                               <114>
2810 PRINT"{RVSON,SPACE}DER SPUR. IN 20 SE
     C.{SPACE,DOWN,SPACE}MUSST DU DEN RAUM
     {4SPACE,DOWN,5SPACE}VERLASSEN."            <086>
2820 FOR H=1 TO 7000:NEXT:RETURN                <170>
2830 PRINT"{CLR,3DOWN,RVSON,SPACE}DAS GEHT
      NUR MIT DEM{SPACE,DOWN,SPACE}SCHLUES
     SEL UND EINEM"                             <112>
2840 PRINT"{DOWN,RVSON}ZUSAETZLICHEN SPEZI
     AL-{DOWN,7SPACE}SCHLUESSEL"                <134>
2850 PRINT"{RVSON,2DOWN,SPACE}ER BEFINDET
     SICH IN{2SPACE,DOWN,4SPACE}EINEM TRES
     OR"                                        <031>
2860 FOR H=1 TO 7000:NEXT:RETURN                <210>
2870 PRINT"{CLR,6DOWN,RVSON,SPACE}IM ABFAL
     LEIMER LIEGT{SPACE,DOWN,6SPACE}EIN ZE
     TTEL{BLACK}"                               <027>
2880 FOR H=1 TO 2000:NEXT:GOSUB 4050:GOTO
     860                                        <173>
2890 PRINT"{CLR,RVSON,3DOWN,2SPACE}DER COD
     E FUER DEN{3SPACE,DOWN,2SPACE}TRESOR
     BESTEHT AUS{2SPACE,DOWN,2SPACE}FOLGEN
     DEN ZAHLEN,"                               <246>
2900 PRINT"{DOWN,RVSON,2SPACE}DIE ABER NOC
     H IN{3SPACE,DOWN,3SPACE}DIE RICHTIGE
     REI-{2SPACE,DOWN,3SPACE}HENFOLGE GEBR
     ACHT"                                      <243>
2910 PRINT"{DOWN,RVSON,2SPACE}WERDEN MUESS
     EN:":PRINT"{3DOWN,6RIGHT,RVSON}";          <159>
2920 W=INT(RND(1)*6)+1                          <235>
2930 ON W GOTO 2940,2950,2960,2970,2980,29
     90                                         <215>
2940 PRINT V3$;V4$;V5$:GOTO 3000                <113>
2950 PRINT V3$;V5$;V4$:GOTO 3000                <108>
2960 PRINT V4$;V3$;V5$:GOTO 3000                <148>
2970 PRINT V4$;V5$;V3$:GOTO 3000                <128>
2980 PRINT V5$;V4$;V3$:GOTO 3000                <153>
2990 PRINT V5$;V3$;V4$:GOTO 3000                <178>
3000 PRINT"{BLACK}":FOR H=1 TO 9000:NEXT:G
     OSUB 4050:GOTO 860                         <085>
3010 FOR H=4097 TO 4096+X:POKE H+F,2:POKE
     H,33:NEXT                                  <116>
3020 GOTO 940                                   <026>
3030 FOR H=4107 TO 4106+X2:POKE H+F,4:POKE
      H,33:NEXT                                 <241>
3040 GOTO 970                                   <094>
3050 FF=FF+1:IF FF=32 THEN FF=24                <082>
3060 RETURN                                     <068>
3070 POKE T,1:POKE T2,250:FOR S=1 TO 20:NE
     XT S:POKE T2,0:POKE T,0:RETURN             <010>
3080 POKE T,5:FOR R=128 TO 255 STEP.2:POKE
      T3,R:NEXT:POKE T,0:POKE T3,0              <039>
3090 POKE 36879,152:PRINT"{CLR}"                <048>
3100 PRINT"{HOME,DOWN,4RIGHT}"O$:PRINT"{16
     RIGHT}"O$:PRINT"{2RIGHT}"O$                <071>
3110 PRINT"{HOME,6DOWN,BLACK,SPACE}↑↑]↑{3R
     IGHT}↑↑]]↑↑↑]"SPC(30)"↑]↑]]↑↑↑"            <021>
3120 PRINT"{4DOWN}"SPC(13)"]]]↑{3RIGHT}↑↑"
     SPC(58)"↑↑↑↑]]↑]"                          <162>
3130 PRINT"{3DOWN,SPACE}↑]{3RIGHT}]]]↑↑]"      <008>
3140 FOR Q=1 TO 2000:NEXT                       <200>
3150 PRINT N$:PRINT                             <045>
3160 FOR P=1 TO 15:PRINT"{RVSON}"MID$(N1$,
     P,1);                                      <155>
3170 O=O+1                                      <225>
3180 IF MID$(N1$,P,1)=CHR$(32)THEN NEXT P      <077>
3190 GOSUB 3220:NEXT P                          <017>
3200 FOR Q=1 TO 800:NEXT:FOR R=4373 TO 436
     0 STEP-1:POKE R,32:NEXT                    <092>
3210 PRINT"{HOME,11DOWN}":GOTO 3160             <112>
3220 IF O>60 THEN PRINT"{CLR}":GOTO 160         <187>
3230 POKE T,2:POKE T3,150:FOR S=1 TO 20:NE
     XT S:POKE T3,0:POKE T,0:RETURN             <214>
3240 S1=4454:S2=4459:S3=4464:S4=4478:F=337
     92:K=4585                                  <124>
3250 IF M1=1 AND F1=1 THEN X=X-4:GOTO 3270 <235>
3260 IF M1=1 THEN X=X-2                         <146>
3270 POKE S1,36:POKE S1+F,0:POKE S2,36:POK
     E S2+F,0:POKE S3,36:POKE S3+F,0            <199>
3280 IF F1=0 THEN 3300                          <106>
3290 POKE S4,36:POKE S4+F,0:POKE K,39:POKE
      K+F,2                                     <108>
3300 POKE T,4:FOR I=170 TO 200:POKE T1,I:N
     EXT:POKE T,0:POKE T1,0                     <161>
```

Listing 2. »Out-Break« (Fortsetzung)

```
3310 IF S1>4580 OR S2>4580 OR S3>4580 OR S
     4>4580 THEN Z=2:GOTO 3750                 <139>
3320 IF F1=0 THEN 3380                         <178>
3330 IF K>4595 THEN 3350                       <112>
3340 IF PEEK(197)=28 THEN POKE K,32:K=K+1:
     POKE K+F,2:POKE K,39                      <155>
3350 IF K<4586 THEN 3370                       <069>
3360 IF PEEK(197)=35 THEN POKE K,32:K=K-1:
     POKE K+F,2:POKE K,39                      <086>
3370 IF PEEK(653)=1 THEN GOSUB 3730:GOSUB
     3610                                      <241>
3380 IF E5=1 THEN E6=E6+1                      <023>
3390 IF E6=90 THEN 3690                        <128>
3400 IF E1>2 AND E2>2 AND E3>2 AND E4>2 TH
     EN E5=1:GOTO 3310                         <160>
3410 J=J+1:IF J=3 THEN J=0:GOTO 3430           <136>
3420 GOTO 3310                                 <154>
3430 S=INT(RND(1)*4)+1                         <205>
3440 ON S GOTO 3450,3490,3530,3570             <187>
3450 IF PEEK(S1)=32 THEN S1=4454:E1=E1+1       <051>
3460 IF E1=5 THEN 3300                         <089>
3470 POKE S1,32:S1=S1+22:POKE S1,36:POKE S
     1+F,0                                     <101>
3480 GOTO 3300                                 <182>
3490 IF PEEK(S2)=32 THEN S2=4478:E2=E2+1       <240>
3500 IF E2=5 THEN 3300                         <145>
3510 POKE S2,32:S2=S2+22:POKE S2,36:POKE S
     2+F,0                                     <217>
3520 GOTO 3300                                 <222>
3530 IF PEEK(S3)=32 THEN S3=4459:E3=E3+1       <153>
3540 IF E3=5 THEN 3300                         <201>
3550 POKE S3,32:S3=S3+22:POKE S3,36:POKE S
     3+F,0                                     <077>
3560 GOTO 3300                                 <006>
3570 IF PEEK(S4)=32 THEN S4=4464:E4=E4+1       <049>
3580 IF E4=5 THEN 3300                         <001>
3590 POKE S4,32:S4=S4+22:POKE S4,36:POKE S
     4+F,0                                     <195>
3600 GOTO 3300                                 <048>
3610 H=K:FOR I=H-22 TO H-132 STEP-22:POKE
     I+F,5:POKE I,40                           <023>
3620 IF PEEK(I-22)=36 THEN GOSUB 3660:GOTO
     3640                                      <116>
3630 NEXT I                                    <158>
3640 FOR J1=H-22 TO H-132 STEP-22:POKE J1+
     F,1:POKE J1,32:NEXT J1                    <089>
3650 RETURN                                    <152>
3660 IF E5=1 THEN RETURN                       <022>
3670 POKE 4455,36:POKE 4455+F,0:POKE 4457,
     36:POKE 4457+F,0:POKE 4462,36:POKE 44
     62+F,0                                    <052>
3680 RETURN                                    <182>
3690 FOR I1=4582 TO 4605:POKE I1,32:NEXT I
     1:POKE 198,0                              <251>
3700 FOR I2=4448 TO 4580:IF PEEK(I2)=36 TH
     EN Z=2:GOTO 3750                          <170>
3710 NEXT I2                                   <017>
3720 E=0:E1=0:E2=0:E3=0:E4=0:E5=0:E6=0:E7=
     0:GOTO 860                                <188>
3730 POKE T,4:FOR H=250 TO 220 STEP-2:POKE
      T2,H:NEXT:POKE T2,0                      <042>
3740 RETURN                                    <242>
3750 FOR H=1 TO 1000:NEXT                      <101>
3760 IF Z=3 THEN 3780                          <070>
3770 POKE 36869,192:POKE 36879,8:POKE 198,
     0                                         <235>
3780 ON Z GOTO 3790,3840,3950                  <196>
3790 PRINT"{CLR,7DOWN}":FOR Q=1 TO LEN(T$)
     :PRINT MID$(T$,Q,1);                      <197>
3800 IF MID$(T$,Q,1)=CHR$(32)THEN NEXT Q       <106>
3810 GOSUB 3230:NEXT Q                         <163>
3820 GOTO 4020                                 <036>
3830 REM                                       <080>
3840 U=U+1                                     <254>
3850 PRINT"{CLR,6DOWN,YELLOW,2SPACE}DU BIS
     T VON EINER{3SPACE,DOWN,3SPACE}SPINNE
      GEBISSEN{4SPACE,DOWN,6SPACE}WORDEN.{
     2DOWN}"                                   <084>
3860 IF G=1 THEN 3890                          <215>
3870 IF U=3 THEN 4020                          <252>
3880 PRINT" DER DRITTE BISS IST{2SPACE,DOW
     N,6SPACE}TOEDLICH."                       <157>
3890 FOR H=1 TO 5000:NEXT                      <245>
3900 PRINT"{CLR,6DOWN,GREEN,SPACE}DURCH DA
     S SPINNENGIFT{DOWN,SPACE}BIST DU VERW
     IRRT UND{SPACE,DOWN,2SPACE}WEISST NIC
     HT WO DU"                                 <140>
3910 PRINT"{DOWN,3SPACE}DICH BEFINDEST.":F
     OR H=1 TO 5000:NEXT:POKE 36869,207:PR
     INT"{CLR,BLACK}"                          <048>
3920 A=INT(RND(1)*17)+1                        <002>
3930 GOSUB 4050:GOTO 860                       <115>
3940 REM                                       <192>
3950 POKE T,15:POKE T4,130                     <211>
3960 FOR H=1 TO 100                            <128>
3970 POKE 36879,124:POKE 36879,143:POKE 36
     879,234:NEXT:POKE T,0:POKE T4,0           <109>
3980 POKE 36879,8:POKE 36869,192:PRINT"{CL
     R,7DOWN}"                                 <119>
3990 FOR Q=1 TO LEN(S$):PRINT MID$(S$,Q,1)
     ;                                         <134>
4000 IF MID$(S$,Q,1)=CHR$(32)THEN NEXT Q       <020>
4010 GOSUB 3230:NEXT Q                         <109>
4020 PRINT:PRINT"{3DOWN,CYAN,4SPACE}- SPIE
     LENDE -{BLACK}"                           <125>
4030 FOR H=1 TO 8000:NEXT:GOTO 160             <218>
4040 END                                       <232>
4050 PRINT"{CLR}":POKE T,2:POKE T2,190:FOR
      H=1 TO 60:NEXT:POKE T,0:POKE T2,0        <189>
4060 C1=C*A                                    <238>
4070 IF A=1 OR A=4 OR A=16 OR A=14 THEN PO
     KE 36879,(FF+160):GOTO 4110               <158>
4080 IF D2=0 AND(C1=13 OR C1=34 OR C1=48)T
     HEN POKE 36879,8:D1=1:GOTO 4110           <202>
4090 IF D2=1 AND(C1=13 OR C1=45 OR C1=34 O
     R C1=48)THEN POKE 36879,152:GOTO 4110     <192>
4100 POKE 36879,FF                             <191>
4110 IF A=1 OR A=2 OR A=4 OR A=8 THEN TY=1     <123>
4120 IF A=3 OR A=11 OR A=14 OR A=16 THEN T
     Y=2                                       <173>
4130 IF A=5 OR A=6 OR A=13 THEN TY=3           <090>
4140 IF A=7 OR A=12 THEN TY=4                  <176>
4150 IF A=10 OR A=15 THEN TY=5                 <062>
4160 IF A=17 THEN TY=6                         <010>
4170 IF A=9 THEN TY=7                          <161>
4180 ON TY GOTO 4190,4220,4300,4380,4410,4
     450,4460                                  <171>
4190 PRINT A1$A2$B1$B2$J1$J2$J3$K1$K2$K3$B
     3$B4$D1$D2$                               <031>
4200 IF(C=2 OR C=3)AND A=8 THEN 3240           <133>
4210 RETURN                                    <204>
4220 PRINT A1$A2$A3$A4$C1$C2$C3$C4$E$          <041>
4230 IF C1=3 AND D=0 THEN PRINT F$             <010>
4240 IF C1=22 AND D=0 THEN PRINT F$            <216>
4250 IF C1=42 AND D=0 THEN PRINT F$            <099>
4260 IF C1=56 AND D=0 THEN PRINT F$            <175>
4270 IF(C=1 OR C=4)AND A=11 AND F1=0 THEN
     PRINT L$                                  <062>
4280 IF(C=2 OR C=3)AND A=3 AND F1=0 THEN P
     RINT L$                                   <020>
4290 RETURN                                    <028>
4300 PRINT A1$A2$A3$A4$B1$B2$B3$B4$I$          <244>
4310 IF A=6 THEN PRINT G1$G2$                  <238>
4320 IF L=0 AND A=5 THEN PRINT H1$             <133>
4330 IF L=3 AND A=6 THEN PRINT H2$             <017>
4340 IF A=6 AND S5=0 THEN POKE 4352+F,2:PO
     KE 4352,35                                <214>
4350 IF G>1 AND A=13 AND T6=0 THEN PRINT V
     $                                         <210>
4360 IF G>1 AND A=13 AND T6=1 THEN PRINT V
     1$                                        <008>
4370 RETURN                                    <110>
4380 PRINT A1$A2$A3$B1$B2$B3$B4$K1$K2$K3$I
     $                                         <105>
4390 IF C=4 AND A=7 THEN 3240                  <216>
4400 RETURN                                    <140>
4410 PRINT A1$A2$A4$J1$J2$J3$B1$B2$B3$B4$D
     1$D2$                                     <076>
4420 IF A=15 AND(M=0 OR M=3)THEN PRINT M$      <163>
4430 IF A=10 AND G>1 THEN PRINT W$             <019>
4440 RETURN                                    <180>
4450 PRINT A1$A2$A3$B1$B2$B3$B4$K1$K2$K3$:
     PRINT"{GREEN}"I$:PRINT"{BLACK}":RETUR
     N                                         <134>
4460 PRINT A1$A2$A4$B1$B2$B3$B4$J1$J2$J3$      <082>
4470 IF C=1 THEN 3240                          <013>
4480 RETURN                                    <220>
```

© 64'er

Listing 2. »Out-Break« (Schluß)

YAATZEE auf dem C16

Wollen Sie ein interessantes Würfelspiel auf Ihrem Computer spielen? Dann schlagen Sie nicht gleich die nächste Seite auf, denn wir haben etwas für Sie.

YAATZEE ist ein Spiel, das viele auch unter dem Namen »KNIFFEL« kennen. Die nachfolgenden Spielregeln zeigen Ihnen, was Sie nach der Eingabe des nebenstehenden Programms (siehe Listing) erwartet. Nun zu den Spielregeln.

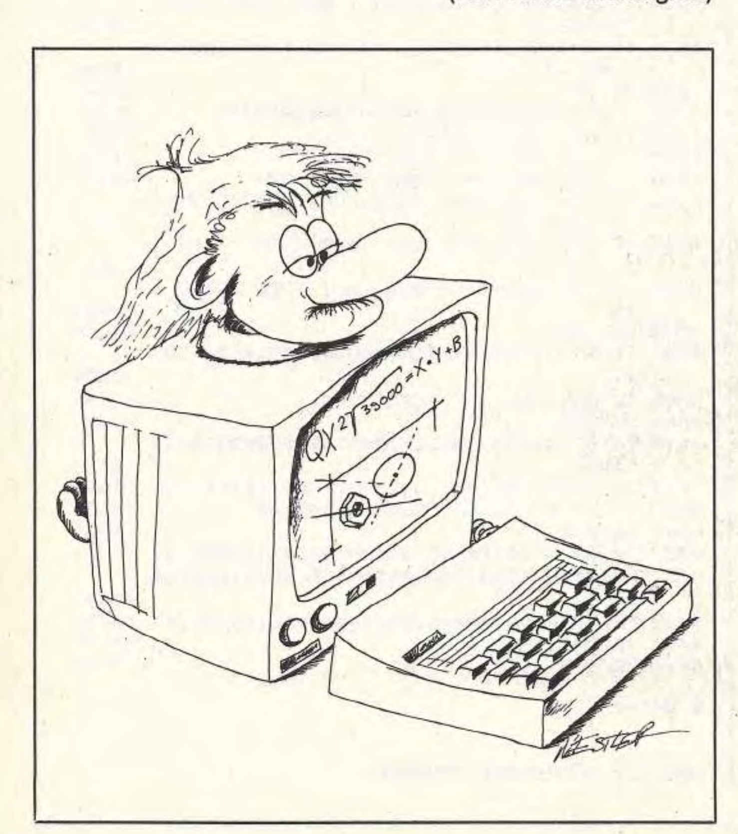
Am Anfang fallen alle 5 Würfel. Davon können 0 bis 5 Würfel getauscht werden, indem Sie die Anzahl der zu tauschenden Würfel eingeben (GETKEY-Abfrage). Daraufhin fragt das Programm ab, welche Würfel getauscht werden sollen. Die getauschten Würfel werden neu geworfen, und Sie haben anschließend erneut die Möglichkeit zum Tausch. Drei Versuche stehen Ihnen zur Verfügung.

Anschließend können Sie mit der Abfrage »WOHIN« Ihren gesamten Wurf einer von 13 Rubriken (Tabelle 1) zuordnen. Diese Rubrik ist dann belegt und kann nicht mehr aufgerufen werden. Hatten Sie keinen Erfolg, müssen Sie trotzdem eine Rubrik belegen. Der Wert 0 wird dort eingetragen.

Wenn Sie bei den Rubriken 1 bis 6 insgesamt mehr als 62 Punkte erreicht haben, erhalten Sie einen Bonus von 35 Punkten.

Wer das Spiel verändern will, findet in Tabelle 2 die Variablenliste und in Tabelle 3 eine Programmübersicht.

(Karl-Heinz Montag/kn)



Mögliche Rubriken	
1) 1er	Hier zählen nur die gefallenen 1er
2) 2er	Hier zählen nur die gefallenen 2er
3) 3er	Hier zählen nur die gefallenen 3er
4) 4er	Hier zählen nur die gefallenen 4er
5) 5er	Hier zählen nur die gefallenen 5er
6) 6er	Hier zählen nur die gefallenen 6er
8) Drilling	3 Würfel mit gleichem Wert, es zählen alle Augen.
9) Vierling	4 Würfel mit gleichem Wert, es zählen alle Augen.
10) Full House	Je 3 und 2 gleiche Würfel, Zählwert 25 Punkte
11) Kleine Straße	4 Würfel in aufsteigender Folge (zum Beispiel 2,3,4,5) Zählwert 30 Punkte.
12) Große Straße	5 Würfel in aufsteigender Folge (zum Beispiel 1,2,3,4,5) Zählwert 40 Punkte)
13) YAATZEE	5 Würfel mit gleichem Wert, Zählwert 50 Punkte
14) Chance	Diese kann man anfordern, wenn keine bestimmte Kombination erzielt wurde oder die erzielte bereits belegt ist. Es werden alle Augen gezählt.

Tabelle 1. Rubriken, denen ein Wurf zugeordnet werden kann

VARIABLENLISTE	
A$(1) bis A$(6)	: 6 mögliche Würfel
A (1) bis A(15)	: mögliche Kombinationen
C (1) bis C(13)	: Überprüfung, ob Kombinationen schon belegt
K	: Zeiger für Spielende
R	: Zähler für 3 Würfe
S	: Zähler für 13 mögliche Kombinationen
AA,AB,AC,AD,AE	: 5 Würfel
BA,BB,BC,BD,BE,BF	: Zähler für Drilling, Vierling und Full House
AG	: Anzahl Würfel tauschen
U(1) bis U(5)	: Würfel tauschen
B,TT	: Warteschleife
O	: Abfrage welche Kombination
V(P),W(P)	: DATAs für Melodie
T	: Highscore

Tabelle 2. Variablenliste zu »YAATZEE«

Zeilen	Funktion
10- 102	Dimensionierung, Variablenbestimmung
103- 300	Bilderstellung
301- 801	Würfeln
900	Überprüfung, ob schon dreimal gewürfelt wurde
1000- 1010	Abfrage wieviele tauschen
1100- 1303	Abfrage welche tauschen
1399- 6402	Würfel tauschen
6510	Eingabe welche Kombinationen belegen
7000- 7005	Zähler für 1er
7100- 7105	Zähler für 2er
7200- 7205	Zähler für 3er
7300- 7305	Zähler für 4er
7400- 7405	Zähler für 5er
7500- 7505	Zähler für 6er
7600	Zwischensumme
7700- 8204	Zähler für Drilling, Vierling und Full House
8300- 8302	Überprüfung ob Drilling
8400- 8402	Überprüfung ob Vierling
8500- 9050	Überprüfung ob Full House
9060- 9110	Überprüfung ob große Straße
9300- 9306	Überprüfung ob YAATZEE
9610- 9693	Melodie
9700- 9720	Highscore
9901- 9910	Abfrage: nochmal
10000-10060	Würfel

Tabelle 3. Programmübersicht zu »YAATZEE«

```
10 DIMA(15),C(15),V(38),W(38) :GOSUB1000
0
20 FORX=1TO15:A(X)=0:C(X)=0:NEXT
90 SCNCLR
91 S=0:VOL3
92 COLOR0,8,7:COLOR4,7,3
100 COLOR1,1:A(7)=A(1)+A(2)+A(3)+A(4)+A(
5)+A(6):IFA(7)>62THENA(7)=A(7)+35
101 GOSUB9450:R=0:K=0:S=S+1:IFS>13THEN95
00
102 SCNCLR:AA=0:AB=0:AC=0:AD=0:AE=0
103 BA=0:BB=0:BC=0:BD=0:BE=0:BF=0
106 PRINT"{5SPACE,RVSON,RED}1{RVOFF,6SPA
CE,RVSON,CYAN}2{RVOFF,6SPACE,RVSON,PURPL
E}3{RVOFF,6SPACE,RVSON,GREEN}4{RVOFF,6SP
ACE,RVSON,BLUE}5{RVSON}"
107 PRINT"{3SPACE}UCCCI  UCCCI  UCCCI  U
CCCI  UCCCI"
110 PRINT"{3SPACE}B{3SPACE}B{2SPACE}B{3S
PACE}B  B{3SPACE}B  B{3SPACE}B{2SPACE}B{
3SPACE}B"
120 PRINT"{3SPACE}B{3SPACE}B  B{3SPACE}B
  B{3SPACE}B  B{3SPACE}B  B{3SPACE}B"
130 PRINT"{3SPACE}B{3SPACE}B  B{3SPACE}B
  B{3SPACE}B  B{3SPACE}B  B{3SPACE}B"
140 PRINT"{3SPACE}JCCCK  JCCCK  JCCCK{2S
PACE}JCCCK  JCCCK"
150 PRINT:
160 PRINT"{3SPACE}1)  1ER{9SPACE}="A(1)
170 PRINT"{3SPACE}2)  2ER{9SPACE}="A(2)
180 PRINT"{3SPACE}3)  3ER{9SPACE}="A(3)
190 PRINT"{3SPACE}4)  4ER{9SPACE}="A(4)
200 PRINT"{3SPACE}5)  5ER{9SPACE}="A(5)
210 PRINT"{3SPACE}6)  6ER{9SPACE}="A(6)
220 PRINT"{4SPACE,PURPLE}ZW-SUMME{6SPACE
}={5SPACE,RVSON}"A(7)"{RVOFF,BLACK}"
230 PRINT"{3SPACE}8)DRILLING{5SPACE}=":I
FC(7)=1THEN231:ELSE232
231 PRINT"{UP,19RIGHT,RVSON,RED}"A(8)"{R
VOFF,BLACK}":GOTO240
232 PRINT"{UP,19RIGHT}"A(8)
240 PRINT"{3SPACE}9)VIERLING{5SPACE}=":I
FC(8)=1THEN241:ELSE242
241 PRINT"{UP,19RIGHT,RVSON,RED}"A(9)"{R
VOFF,BLACK}":GOTO250
242 PRINT"{UP,19RIGHT}"A(9)
250 PRINT"{3SPACE}10)FULLHOUSE{3SPACE}="
:IFC(9)=1THEN251:ELSE252
251 PRINT"{UP,19RIGHT,RVSON,RED}"A(10)"{
RVOFF,BLACK}":GOTO260
252 PRINT"{UP,19RIGHT}"A(10)
260 PRINT"{3SPACE}11)KL.STRASSE{2SPACE}=
":IFC(10)=1THEN261:ELSE262
261 PRINT"{UP,19RIGHT,RVSON,RED}"A(11)"{
RVOFF,BLACK}":GOTO270
262 PRINT"{UP,19RIGHT}"A(11)
270 PRINT"{3SPACE}12)GR.STRASSE{2SPACE}=
":IFC(11)=1THEN271:ELSE272
271 PRINT"{UP,19RIGHT,RVSON,RED}"A(12)"{
RVOFF,BLACK}":GOTO280
272 PRINT"{UP,19RIGHT}"A(12)
280 PRINT"{3SPACE}13)YAATZEE{5SPACE}=":I
FC(12)=1THEN281:ELSE282
281 PRINT"{UP,19RIGHT,RVSON,RED}"A(13)"{
RVOFF,BLACK}":GOTO290
282 PRINT"{UP,19RIGHT}"A(13)
290 PRINT"{3SPACE}14)CHANCE{6SPACE}=":IF
C(13)=1THEN291:ELSE292
291 PRINT"{UP,19RIGHT,RVSON,RED}"A(14)"{
RVOFF,BLACK}":GOTO300
292 PRINT"{UP,19RIGHT}"A(14)
300 PRINT"{5SPACE,GREEN}GESAMT{7SPACE}={
5SPACE,RVSON}"A(15)"{RVOFF,BLACK}"
301 IFK=100THEN9600
400 AA=INT(RND(1)*6)+1
401 PRINT"{HOME,2DOWN,4RIGHT}"A$(AA)
500 AB=INT(RND(1)*6)+1
501 PRINT"{HOME,2DOWN,11RIGHT}"A$(AB)
600 AC=INT(RND(1)*6)+1
601 PRINT"{HOME,2DOWN,18RIGHT}"A$(AC)
700 AD=INT(RND(1)*6)+1
701 PRINT"{HOME,2DOWN,25RIGHT}"A$(AD)
800 AE=INT(RND(1)*6)+1
801 PRINT"{HOME,2DOWN,32RIGHT}"A$(AE)
900 R=R+1:IFR<3THEN990 :ELSE6500
990 PRINT"{HOME,23DOWN,30SPACE}"
1000 PRINT"{HOME,22DOWN}WIEVIELE WILLST
DU TAUSCHEN?"
1010 GETKEYAG:IFAG=0THEN6500:IFAG=7THEN7
600
1020 PRINT"{HOME,23DOWN,7SPACE}"
1100 PRINT"{HOME,23DOWN}WELCHE"
1200 FORAF=1TOAG
1300 GETKEYU(AF):U=AF:SOUND1, 99,6 :NEXT
1301 IFU     >0ANDU     <6THEN1399:ELSE130
2
1302 PRINT"{HOME,23DOWN,6SPACE}"
1303 PRINT"{HOME,23DOWN}WELCHE":GOTO1200
1399 FORB=1TO500:NEXT
1400 ONU(1)GOSUB6000,6100,6200,6300,6400
1409 FORB=1TO500:NEXT
1410 ONU(2)GOSUB6000,6100,6200,6300,6400
1419 FORB=1TO500:NEXT
1420 ONU(3)GOSUB6000,6100,6200,6300,6400
1429 FORB=1TO500:NEXT
1430 ONU(4)GOSUB6000,6100,6200,6300,6400
1439 FORB=1TO500:NEXT
1440 ONU(5)GOSUB6000,6100,6200,6300,6400
1450 FORAF=1TO5:U(AF)=0:NEXT
1500 GOTO900
6000 PRINT"{HOME,2DOWN,4RIGHT,3SPACE,DOW
N,3LEFT,3SPACE,DOWN,3LEFT,3SPACE}"
6001 AA=INT(RND(1)*6)+1
6002 PRINT"{HOME,2DOWN,4RIGHT}"A$(AA):GO
SUB11000:RETURN
6100 PRINT"{HOME,2DOWN,11RIGHT,3SPACE,DO
WN,3LEFT,3SPACE,DOWN,3LEFT,3SPACE}"
6101 AB=INT(RND(1)*6)+1
6102 PRINT"{HOME,2DOWN,11RIGHT}"A$(AB):G
OSUB11000:RETURN
6200 PRINT"{HOME,2DOWN,18RIGHT,3SPACE,DO
WN,3LEFT,3SPACE,DOWN,3LEFT,3SPACE}"
6201 AC=INT(RND(1)*6)+1
6202 PRINT"{HOME,2DOWN,18RIGHT}"A$(AC):G
OSUB11000:RETURN
6300 PRINT"{HOME,2DOWN,25RIGHT,3SPACE,DO
WN,3LEFT,3SPACE,DOWN,3LEFT,3SPACE}"
6301 AD=INT(RND(1)*6)+1
6302 PRINT"{HOME,2DOWN,25RIGHT}"A$(AD):G
OSUB11000:RETURN
6400 PRINT"{HOME,2DOWN,32RIGHT,3SPACE,DO
WN,3LEFT,3SPACE,DOWN,3LEFT,3SPACE}"
6401 AE=INT(RND(1)*6)+1
6402 PRINT"{HOME,2DOWN,32RIGHT}"A$(AE):G
OSUB11000:RETURN
6500 PRINT"{HOME,22DOWN,31SPACE}"
6501 PRINT"{HOME,23DOWN}WOHIN{2SPACE}"
6510 INPUTO:GOTO6520
6511 PRINT"{HOME,22DOWN,31SPACE}"
6512 PRINT"{HOME,21DOWN}WOHIN "
6513 INPUTO
6520 IFO=1THEN7000:ELSE6530
6530 IFO=2THEN7100:ELSE6540
6540 IFO=3THEN7200:ELSE6550
6550 IFO=4THEN7300:ELSE6560
6560 IFO=5THEN7400:ELSE6570
6570 IFO=6THEN7500:ELSE6580
6580 IFO=8THEN7700:ELSE6590
6590 IFO=9THEN7700:ELSE6600
6600 IFO=10THEN7700:ELSE6610
6610 IFO=11THEN9060:ELSE6620
6620 IFO=12THEN9095:ELSE6630
6630 IFO=13THEN9300:ELSE6640
6640 IFO=14THEN9400:ELSE6650
6650 GOTO100
7000 IFC(1)=1THEN6511:A(1)=0:ELSEIFAA=1T
HENA(1)=A(1)+1:ELSEAA=0
7001 IFAB=1THENA(1)=A(1)+1:ELSEAB=0
7002 IFAC=1THENA(1)=A(1)+1:ELSEAC=0
7003 IFAD=1THENA(1)=A(1)+1:ELSEAD=0
7004 IFAE=1THENA(1)=A(1)+1:ELSEAE=0:C(1)
=1
7005 GOTO100
7100 IFC(2)=1THEN6511:A(2)=0:ELSEIFAA=2T
HENA(2)=A(2)+2:ELSEAA=0
7101 IFAB=2THENA(2)=A(2)+2:ELSEAB=0
7102 IFAC=2THENA(2)=A(2)+2:ELSEAC=0
7103 IFAD=2THENA(2)=A(2)+2:ELSEAD=0
7104 IFAE=2THENA(2)=A(2)+2:ELSEAE=0:C(2)
=1
7105 GOTO100
```

Listing zu »YAATZEE«

```
7200 IFC(3)=1THEN6511:A(3)=0:ELSEIFAA=3T
HENA(3)=A(3)+3:ELSEAA=0
7201 IFAB=3THENA(3)=A(3)+3:ELSEAB=0
7202 IFAC=3THENA(3)=A(3)+3:ELSEAC=0
7203 IFAD=3THENA(3)=A(3)+3:ELSEAD=0
7204 IFAE=3THENA(3)=A(3)+3:ELSEAE=0:C(3)
=1
7205 GOTO100
7300 IFC(4)=1THEN6511:A(4)=0:ELSEIFAA=4T
HENA(4)=A(4)+4:ELSEAA=0
7301 IFAB=4THENA(4)=A(4)+4:ELSEAB=0
7302 IFAC=4THENA(4)=A(4)+4:ELSEAC=0
7303 IFAD=4THENA(4)=A(4)+4:ELSEAD=0
7304 IFAE=4THENA(4)=A(4)+4:ELSEAE=0:C(4)
=1
7305 GOTO100
7400 IFC(5)=1THEN6511:A(5)=0:ELSEIFAA=5T
HENA(5)=A(5)+5:ELSEAA=0
7401 IFAB=5THENA(5)=A(5)+5:ELSEAB=0
7402 IFAC=5THENA(5)=A(5)+5:ELSEAC=0
7403 IFAD=5THENA(5)=A(5)+5:ELSEAD=0
7404 IFAE=5THENA(5)=A(5)+5:ELSEAE=0:C(5)
=1
7405 GOTO100
7500 IFC(6)=1THEN6511:A(6)=0:ELSEIFAA=6T
HENA(6)=A(6)+6:ELSEAA=0
7501 IFAB=6THENA(6)=A(6)+6:ELSEAB=0
7502 IFAC=6THENA(6)=A(6)+6:ELSEAC=0
7503 IFAD=6THENA(6)=A(6)+6:ELSEAD=0
7504 IFAE=6THENA(6)=A(6)+6:ELSEAE=0:C(6)
=1
7505 GOTO100
7600 A(7)=A(1)+A(2)+A(3)+A(4)+A(5)+A(6)
7601 IFA(7)>62THENA(7)=A(7)+35:GOTO100
7700 IFAA=1THENBA=BA+1
7701 IFAB=1THENBA=BA+1
7702 IFAC=1THENBA=BA+1
7703 IFAD=1THENBA=BA+1
7704 IFAE=1THENBA=BA+1
7800 IFAA=2THENBB=BB+1
7801 IFAB=2THENBB=BB+1
7802 IFAC=2THENBB=BB+1
7803 IFAD=2THENBB=BB+1
7804 IFAE=2THENBB=BB+1
7900 IFAA=3THENBC=BC+1
7901 IFAB=3THENBC=BC+1
7902 IFAC=3THENBC=BC+1
7903 IFAD=3THENBC=BC+1
7904 IFAE=3THENBC=BC+1
8000 IFAA=4THENBD=BD+1
8001 IFAB=4THENBD=BD+1
8002 IFAC=4THENBD=BD+1
8003 IFAD=4THENBD=BD+1
8004 IFAE=4THENBD=BD+1
8100 IFAA=5THENBE=BE+1
8101 IFAB=5THENBE=BE+1
8102 IFAC=5THENBE=BE+1
8103 IFAD=5THENBE=BE+1
8104 IFAE=5THENBE=BE+1
8200 IFAA=6THENBF=BF+1
8201 IFAB=6THENBF=BF+1
8202 IFAC=6THENBF=BF+1
8203 IFAD=6THENBF=BF+1
8204 IFAE=6THENBF=BF+1
8205 IFO=10THEN8500:ELSE8206
8206 IFO=9THEN8400:ELSE8207
8207 IFO=8THEN8300
8300 IFC(7)=1THEN6511:ELSE8301
8301 IFBA>2ORBB>2ORBC>2ORBD>2ORBE>2ORBF>
2THEN8302:ELSEC(7)=1:A(8)=0:GOTO100
8302 A(8)=AA+AB+AC+AD+AE:C(7)=1:GOTO100
8400 IFC(8)=1THEN6511:ELSEIFBA>3ORBB>3OR
BC>3ORBD>3ORBE>3ORBF>3THEN8402
8401 C(8)=1:A(9)=0:GOTO100
8402 A(9)=AA+AB+AC+AD+AE:C(8)=1:GOTO100
8500 IFC(9)=1THEN6511:ELSEIFBA=3THEN8501
:ELSE8600
8501 IFBB=2ORBC=2ORBD=2ORBE=2ORBF=2THEN9
050:ELSE8600
8600 IFBB=3THEN8601:ELSE8700
8601 IFBA=2ORBC=2ORBD=2ORBE=2ORBF=2THEN9
050:ELSE8700
8700 IFBC=3THEN8701:ELSE8800
8701 IFBA=2ORBB=2ORBD=2ORBE=2ORBF=2THEN9
050:ELSE8800
8800 IFBD=3THEN8801:ELSE8900
8801 IFBA=2ORBB=2ORBC=2ORBE=2ORBF=2THEN9
050:ELSE8900
8900 IFBE=3THEN8901:ELSE9000
8901 IFBA=2ORBB=2ORBC=2ORBD=2ORBF=2THEN9
050:ELSE9000
9000 IFBF=3THEN9001:ELSE9020
9001 IFBA=2ORBB=2ORBC=2ORBD=2ORBE=2THEN9
050:ELSE9020
9020 A(10)=0:C(9)=1:GOTO100
9050 A(10)=25:C(9)=1:GOTO100
9060 IFC(10)=1THEN6511:ELSEA(11)=30:C(10
)=1:GOTO100
9095 IFC(11)=1THEN6511:ELSE9098
9098 IFAA<>ABANDAA<>ACANDAA<>ADANDAA<>AE
THEN9102:ELSE9108
9102 IFAB<>ACANDAB<>ADANDAB<>AEANDAB<>AA
THEN9103:ELSE9108
9103 IFAC<>AAANDAC<>ABANDAC<>ADANDAC<>AE
THEN9104:ELSE9108
9104 IFAD<>AAANDAD<>ABANDAD<>ACANDAD<>AE
THEN9105:ELSE9108
9105 IFAE<>AAANDAE<>ABANDAE<>ACANDAE<>AD
THEN9106:ELSE9108
9106 IFAA>1ANDAB>1ANDAC>1ANDAD>1ANDAE>1T
HEN9110:ELSE9107
9107 IFAA<6ANDAB<6ANDAC<6ANDAD<6ANDAE<6T
HEN9110:ELSE9108
9108 A(12)=0:C(11)=1:GOTO100
9110 A(12)=40:C(11)=1:GOTO100
9300 IFC(12)=1THEN6511:ELSE9302
9302 IFAA=ABANDAA=ACANDAA=ADANDAA=AETHEN
9306:ELSE9305
9305 A(13)=0:C(12)=1:GOTO100
9306 A(13)=50:C(12)=1:GOTO100
9400 IFC(13)=1THEN6511:ELSEA(14)=AA+AB+A
C+AD+AE:C(13)=1:GOTO100
9450 A(15)=A(7)+A(8)+A(9)+A(10)+A(11)+A(
12)+A(13)+A(14):RETURN
9500 A(15)=A(7)+A(8)+A(9)+A(10)+A(11)+A(
12)+A(13)+A(14):K=100:GOTO103
9600 FORTT=1TO3000:NEXT
9610 IFA(15)>290THEN9620:ELSE9700
9620 SCNCLR:FORP=1TO38
9630 READV(P),W(P)
9640 SOUND1,V(P),W(P)
9650 PRINT"{RVOFF,RED}S{GREEN,RVSON}"A(1
5)"{RVOFF,LEFT,BLUE}SS";"{RED}S{GREEN,RV
SON}"A(15)"{RVOFF,LEFT,RED}SS";"{RED}S{G
REEN,RVSON}"A(15)"{RVOFF,LEFT,RED}SS";"{
RED}S{PURPLE,RVSON}"A(15)"{RVOFF,LEFT,RE
D}SS";
9660 NEXTP
9670 DATA 643,20,739,20,739,10,770,10,79
8,20,739,20,834,40
9680 DATA 798,30,798,10,810,20,834,10,81
0,10,798,10,810,10
9690 DATA 834,20,770,10,739,10,770,10,79
8,10,770,20
9691 DATA1021,2 ,643,20,739,20,739,10,77
0,10,798,20,739,20,834,40,798,30
9692 DATA798,10,810,10,834,10,798,10,810
,10,770,30,739,10,1021,2,739,30
9693 RESTORE
9695 FORTT=1TO3000:NEXT
9700 IFA(15)>TTHENT=A(15):GOTO9710:ELSE9
900
9710 COLOR1,2:COLOR0,1:COLOR4,1
9720 PRINT"{CLR,9RIGHT,9DOWN}HIGHSCORE:{
RVSON}"T"{RVOFF}"
9900 FORTT=1TO3000:NEXT
9901 PRINT"{3DOWN}NOCHMAL (J/N)?"
9910 GETKEYA$:IFA$="J"THEN20:ELSEEND
10000 A$(1)="{RIGHT,DOWN}Q"
10010 A$(2)="{RED}Q{2DOWN,RIGHT}Q{BLACK}
"
10020 A$(3)="{CYAN}Q{DOWN}Q{DOWN}Q{BLACK
}"
10030 A$(4)="{PURPLE}Q{RIGHT}Q{DOWN,3LEF
T,DOWN}Q{RIGHT}Q{BLACK}"
10040 A$(5)="{GREEN}Q{RIGHT}Q{DOWN,2LEFT
}Q{DOWN,2LEFT}Q{RIGHT}Q{BLACK}"
10050 A$(6)="{BLUE}Q{RIGHT}Q{DOWN,3LEFT}
Q{RIGHT}Q{DOWN,3LEFT}Q{RIGHT}Q{BLACK}"
10060 RETURN
11000 SOUND1,1000,6:RETURN
(C) 64'ER
```

Listing zu »YAATZEE« (Schluß)

LIFE – Das Spiel des Lebens

Mit diesem Spiel erforschen Sie das Leben. Organismen werden geboren, überleben oder sterben im Evolutionsprozeß. Wie sich die Organismen entwickeln, liegt an Ihnen.

Das Spiel »LIFE« besteht aus zwei Programmteilen, einem Maschinenspracheteil »Life mc loader« (Listing 1) und dem Hauptprogramm »Life basic program« (Listing 2). Für das Spiel wird eine 3 KByte Erweiterung benötigt. Haben Sie beide Programmteile abgetippt und gespeichert, ist zunächst » life mc loader« zu laden und mit RUN zu starten. Geben Sie nun NEW ein und laden das Hauptprogramm, das sich ebenfalls mit RUN starten läßt.

Zum Programm:

»LIFE« könnte man als mathematisches »Spiel« bezeichnen. Es ist geradezu prädestiniert für die Implementierung auf Computern. Entwickelt wurde es vom britischen Mathematiker Arthur Conway. Mit »LIFE« werden in einer Matrix geometrische Figuren entwickelt, die, und das ist die Besonderheit, in ihrer Entwicklung lebenden primitiven Organismen ähneln. Als Ausgangsfigur legt man in einer Matrix mit Reihen und Spalten ein beliebiges Muster fest. Dieses wird nach den folgenden Regeln fortentwickelt:

Ein Feld in der Matrix hat genau acht Nachbarfelder. Befindet sich in dem Feld nun eine »Zelle«, so wird darüber entschieden, ob diese Zelle »überlebt« oder »stirbt«. Existieren auf den Nachbarfeldern dieser Zelle zwei oder drei weitere Zellen, so »überlebt« die Zelle. Befinden sich dort weniger als zwei oder mehr als drei, so »stirbt« sie. Auf einem freien Feld, das genau drei Nachbarfelder mit Zellen hat, wird eine neue Zelle »geboren«. Diese Regeln werden alle simultan angewandt, das heißt unabhängig davon, ob eine Zelle »stirbt«, »überlebt« oder »geboren« wird, wird für die übrigen Zellen das noch bestehende Muster für die Anwendung der Regeln herangezogen. Ein in oben genannter Weise neu entstandener »Organismus« ist ein solcher der »zweiten Generation«. Diesen kann man nun genauso wie den ersten fortentwickeln.

Dieses ist schon vom Optischen her sehr reizvoll. Es können zum Beispiel aus unscheinbaren Ausgangsmustern aufgeblähte Konfigurationen entstehen, die aber ebenso plötzlich in sich zusammenfallen können. Es existieren pulsierende Muster, die sich nach einer gewissen Zeitspanne wiederholen.

Um ein wenig »Evolution« in diese Regeln zu bringen, ermöglicht das Programm auch, zufällig eine bestimmte Anzahl von Zellen zu »zerstören«.

Bedienung des Programms:

Das Programm ist menügesteuert. Am Anfang läßt sich ein Grundmuster eingeben. Dazu wählt man »Option 1« (Entering of Coordinates). Falls man schon vorher ein Muster eingegeben hat, kann man wählen, ob dieses vor Neueingabe gelöscht werden soll oder nicht. Danach erscheint auf dem Bildschirm eine Matrix aus 32 Zeilen und 26 Spalten mit einem roten Cursor in der Mitte. Diesen kann man wie den »normalen« Cursor mit den Steuertasten bewegen. Will man an einer Stelle eine »Zelle« setzen, so drückt man die Taste »*«, will man eine vorhandene Zelle löschen, so drückt man die Taste SPACE. Mit der Taste »R« kommt man ins Menü zurück.

Nun läßt sich die Figur fortentwickeln.

Dazu gibt es zwei Optionen. Die »Automatic«-Option entwickelt sich von selbst. Ist ein »Organismus« aufgebaut, so wird gleich darauf der der nächsten Generation erstellt. Im »Single Step Mode« wartet das Programm jeweils, bis die SPACE-Taste gedrückt wird. Erst dann wird der nachfolgende Organismus dargestellt. In beiden Modi kann man mit der Taste »F3« die Hintergrundfarbe und mit »F5« die Zeichenfarbe beliebig ändern. Mit »F7« kommt man ins Menü zurück. Die Tasten müssen einige Sekunden gedrückt werden. Grafisch ist das besonders reizvoll. Zusätzlich läßt sich auf einem Drucker eine Hardcopy erzeugen, wenn man auf die Frage mit Y (Yes) antwortet. Diese wird so erstellt, daß überflüssige Zeilen, die kein Muster enthalten, nicht gedruckt werden.

Schließlich kann man mit Option 4 »Mutationen« erzeugen. Dazu wird nach der relativen Häufigkeit gefragt. »1« ist die geringste Häufigkeit, nach oben ist keine Grenze gesetzt. Mit Option 5 läßt sich die Mutation wieder »ausschalten«. – Interessant sind zum Beispiel Studien, ob Mutationen der »Entwicklung« förderlich oder für sie eher schädlich sind, oder über die Tatsache, daß der Trend der »Entwicklung« meist vom großen zu mehreren kleinen Organismen geht, in die sich der Große aufspaltet.

(Claus Neupert/ah)

```
1 POKE 55,120:POKE 56,23                          <253>
10 DIM H(75):FOR I=0 TO 9                         <139>
20 H(48+I)=I:H(65+I)=I+10:NEXT                    <067>
30 FOR I=6010 TO 7167:READ A$                     <004>
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))         <223>
50 D=H(H)*16+H(L):S=S+D:POKE I,D                  <243>
60 A=A+1:IF A<20 THEN NEXT: A=-1                  <165>
65 PRINT"ZEILE ";1000+Z;                          <181>
70 READ V:Z=Z+1:IF V=S THEN 85                    <049>
80 PRINT" FEHLER !";999+Z:STOP                    <023>
85 IF A<0 THEN END                                <247>
90 S=0:A=0:PRINT:NEXT:END                         <153>
1000 DATA 32,42,31,41,30,00,00,00,00,00,A9
     ,00,8D,DB,1B,8D,DC,1B,A9,08, 1399            <152>
1001 DATA 8D,D8,1B,4U,D9,1B,90,12,18,AT,DA
     ,1B,6D,DB,1B,8D,DB,1B,A9,00, 2221            <136>
1002 DATA 6D,DC,1B,8D,DC,1B,4E,DB,1B,6E,DC
     ,1B,CE,D8,1B,D0,DE,60,EA,EA, 2868            <173>
1003 DATA AD,CE,1B,18,6D,D0,1B,8D,D2,1B,AD
     ,CD,1B,18,6D,CF,1B,90,03,EE, 2309            <147>
1004 DATA D2,1B,8D,D1,1B,60,EA,EA,EA,EA,AD
     ,D7,1B,8D,D9,1B,A9,1B,8D,DA, 3001            <221>
1005 DATA 1B,20,84,17,AD,DC,1B,8D,CD,1B,AD
     ,DB,1B,8D,CE,1B,AD,D6,1B,8D, 2355            <213>
1006 DATA CF,1B,A9,00,8D,D0,1B,20,B6,17,AD
     ,D2,1B,18,6D,CC,1B,8D,D2,1B, 2168            <067>
1007 DATA 60,A9,00,8D,F8,1B,8D,F9,1B,EA,A9
     ,14,8D,CC,1B,A2,00,A0,00,A9, 2384            <056>
1008 DATA 08,8D,D3,1B,A9,00,8D,DE,1B,8A,18
     ,69,01,8D,D6,1B,98,18,69,01, 1878            <133>
1009 DATA 8D,D7,1B,20,D4,17,AD,D1,1B,8D,41
     ,18,AD,D2,1B,8D,42,18,AD,5E, 2197            <132>
1010 DATA 17,C9,00,D0,05,A9,01,8D,DE,1B,A9
     ,00,8D,D4,1B,CE,D3,1B,8E,D5, 2345            <188>
1011 DATA 1B,AE,D3,1B,BD,BC,1B,18,6D,D5,1B
     ,8D,D6,1B,AE,D5,1B,8C,D5,1B, 2392            <079>
1012 DATA AC,D3,1B,B9,C4,1B,18,6D,D5,1B,8D
     ,D7,1B,AC,D5,1B,20,D4,17,AD, 2426            <140>
1013 DATA D1,1B,8D,8A,18,AD,D2,1B,8D,8B,18
     ,AD,42,17,C9,00,F0,07,C9,FF, 2419            <212>
1014 DATA F0,03,EE,D4,1B,AD,D3,1B,C9,00,D0
     ,B3,AD,DE,1B,C9,00,D0,22,A9, 2753            <053>
1015 DATA 02,CD,D4,1B,F0,18,A9,03,CD,D4,1B
     ,F0,11,AD,41,18,8D,C2,18,AD, 2377            <114>
1016 DATA 42,18,8D,C3,18,A9,02,8D,B9,16,4C
```

Listing 1. Maschinenprogrammteil zu »LIFE«

```
     ,DF,18,A9,03,CD,D4,1B,D0,11, 2133     <241>
1017 DATA AD,41,18,8D,DD,18,AD,42,18,8D,DE
     ,18,A9,FF,8D,C6,15,C8,C0,1F, 2505     <105>
1018 DATA D0,03,4C,EA,18,4C,19,18,A0,00,E8
     ,E0,19,D0,03,4C,F7,18,4C,19, 1970     <078>
1019 DATA 18,4C,00,19,13,37,23,96,63,1A,A2
     ,01,A0,01,A9,14,8D,CC,1B,8E, 1536     <195>
1020 DATA D6,1B,8C,D7,1B,20,D4,17,AD,D1,1B
     ,8D,1F,19,AD,D2,1B,8D,20,19, 2104     <133>
1021 DATA AD,7A,17,18,C9,00,F0,15,C9,02,D0
     ,08,A9,00;20,50,19,4C,3B,19, 1689     <127>
1022 DATA C9,FF,EA,EA,A9,01,20,26,1B,C8,C0
     ,21,D0,C4,A0,01,E8,E0,1B,D0, 2872     <231>
1023 DATA BD,60,EA,EA,EA,EA,EA,EA,EA,EA,8D
     ,DD,1B,8E,D6,1B,8C,D7,1B,20, 3343     <080>
1024 DATA D4,17,AD,D1,1B,8D,6C,19,AD,D2,1B
     ,8D,6D,19,AD,DD,1B,8D,E3,15, 2413     <209>
1025 DATA C9,01,D0,05,A9,2A,4C,C8,19,A9,20
     ,4C,C8,19,00,8D,DF,1B,4C,C8, 2096     <105>
1026 DATA 19,C9,FF,F0,EB,C9,02,D0,10,A9,20
     ,8D,DF,1B,A9,1C,8D,CC,1B,20, 2576     <102>
1027 DATA D4,17,4C,C8,19,60,8E,C0,CC,CE,CC
     ,CB,CF,56,84,EC,FD,A4,4C,88, 3073     <077>
1028 DATA CC,E0,EC,A5,59,08,31,B2,33,21,33
     ,13,33,33,31,B0,33,13,33,7B, 1878     <193>
1029 DATA 33,37,B9,1B,33,81,33,70,73,5B,8D
     ,E1,1B,AD,CC,1B,8D,E0,1B,A9, 2225     <000>
1030 DATA 1C,8D,CC,1B,20,D4,17,AD,D1,1B,8D
     ,E9,19,AD,D2,1B,8D,EA,19,AD, 2458     <147>
1031 DATA E1,1B,8D,E3,1D,AD,E0,1B,8D,CC,1B
     ,60,33,03,B3,66,27,31,33,13, 2034     <060>
1032 DATA B3,F1,93,33,32,65,C3,43,AB,31,2B
     ,19,AB,91,72,60,73,92,32,42, 2222     <165>
1033 DATA 39,CA,CC,C5,8C,CC,6C,CC,EC,8C,DC
     ,C8,CC,48,CC,CC,CC,CC,CD,94, 3557     <250>
1034 DATA 46,CF,48,CE,4C,6B,4F,DC,45,9E,0C
     ,C6,EA,CC,33,2B,7B,71,93,63, 2488     <157>
1035 DATA 32,39,2B,13,33,37,33,64,93,30,39
     ,B1,27,9C,53,7C,33,D9,37,00, 1580     <021>
1036 DATA 13,03,3B,1A,3B,00,CC,8C,C8,C9,DC
     ,EC,CC,CC,C6,C4,4C,CC,CE,8C, 2795     <189>
1037 DATA DC,EC,CC,2C,4E,43,91,90,4E,8E,CC
     ,48,75,AC,A8,35,4C,9C,33,B3, 2606     <191>
1038 DATA 33,1F,37,23,13,12,33,33,33,31,33
     ,1A,39,F1,81,0B,67,33,33,96, 1281     <086>
1039 DATA 67,2A,B8,13,33,F4,39,B0,36,96,C8
     ,EC,CC,CC,CC,CC,DC,C2,8C,C4, 3082     <079>
1040 DATA EE,C0,4F,CC,CC,CC,4C,E4,ED,EC,DE
     ,ED,E8,93,EC,28,4E,7D,EC,8E, 3593     <202>
1041 DATA 04,E8,3B,33,71,53,33,D3,23,11,33
     ,23,33,F4,33,33,2B,B3,E9,E5, 2023     <255>
1042 DATA 33,70,52,FC,B3,BC,78,79,02,B2,13
     ,8F,96,8F,CC,CE,CC,E7,C4,DC, 3001     <218>
1043 DATA CC,CC,CC,CC,4C,C4,C0,C4,CC,CD,CC
     ,FF,CD,A4,8C,8F,66,CB,C7,ED, 3737     <210>
1044 DATA C9,6D,EC,99,14,CD,11,23,73,31,33
     ,3A,A1,33,13,31,33,37,71,7B, 1871     <175>
1045 DATA B1,3A,13,B7,33,02,2C,C2,13,29,3B
     ,70,27,6D,02,FF,BB,68,CC,84,01991     <225>
1046 DATA CC,DC,D8,DD,8C,4C,CC,DC,CC,CC,DC
     ,DC,CC,CC,8C,D6,CC,DC,4E,E7, 3844     <020>
1047 DATA 18,A9,01,6D,F8,1B,90,03,EE,F9,1B
     ,8D,F8,1B,A9,01,4C,50,19,63, 2105     <102>
1048 DATA 93,B1,B3,77,33,33,66,66,B7,C7,33
     ,59,AB,2B,13,1C,33,13,73,A3, 2059     <053>
1049 DATA 3B,84,8C,54,C8,C4,CF,5C,EC,CC,4C
     ,CC,CC,C8,C5,EC,CE,CC,8C,D4, 3429     <009>
1050 DATA E6,28,8C,DC,4C,84,5D,9D,EC,E8,CA
     ,5A,6E,EE,30,1B,3B,37,31,27, 2473     <112>
1051 DATA B3,F3,B3,7E,33,77,33,77,37,A9,E4
     ,A6,13,89,22,BC,31,01,3B,13, 2191     <076>
1052 DATA 38,1B,1B,A3,B7,E7,8C,6D,D4,CD,CC
     ,C5,CC,ED,DC,CC,CE,EE,C4,CC, 3463     <202>
1053 DATA 4C,C8,8D,EC,D7,C9,C4,4E,CC,DE,CB
     ,CD,CE,A7,85,96,C0,80,37,77, 3327     <125>
1054 DATA 13,2B,17,B5,B3,33,32,97,33,15,00
     ,01,02,00,02,00,01,02,00,00, 777      <216>
1055 DATA 00,01,01,02,02,02,14,60,03,1A,00
     ,7A,17,00,00,1E,1A,20,00,00, 386      <036>
1056 DATA 1B,03,60,01,01,EC,14,2A,C9,1C,C5
     ,DA,CD,DE,D8,EC,DC,88,46,25, 2412     <127>
1057 DATA 3C,8C,B3,21,33,32,73,32,33,02,70
     ,00,73,FF,32,8B,33,33, 1504           <131>
```

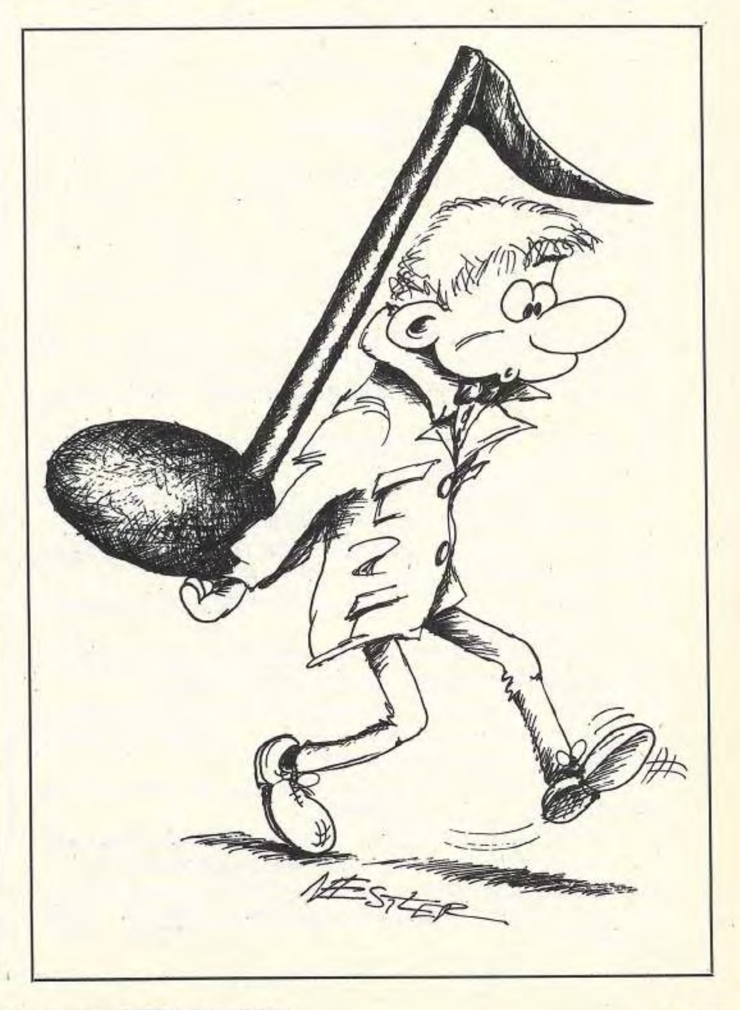

Listing 1. Maschinenprogrammteil zu »LIFE« (Schluß)

```
20 POKE 55,240:POKE 56,19:CLR                   <135>
22 PRINT"{CLR}":POKE 36879,75:POKE 36878,1
   5:T1=234:T2=230:T3=223:                      <180>
26 FOR I=1 TO 10:GOSUB 31:FOR J=1 TO 75 :N
   EXT:POKE 36878,0                             <183>
27 PRINT"{CLR}":FOR J=1 TO 71:NEXT:POKE 36
   878,15:NEXT:POKE 36874,0:POKE 36875,0:P
   OKE 36876,0                                  <250>
28 GOSUB 31:POKE 36874,0:POKE 36875,0:POKE
    36876,0:GOSUB 4000::GOTO 40                 <061>
31 PRINT"{WHITE,HOME,3DOWN,4RIGHT,RVSON,SP
   ACE,DOWN,LEFT,SPACE,DOWN,LEFT,SPACE,DOW
   N,LEFT,SPACE,DOWN,LEFT,3SPACE}"              <168>
32 PRINT"{HOME,3DOWN,8RIGHT,RVSON,SPACE,DO
   WN,LEFT,SPACE,DOWN,LEFT,SPACE,DOWN,LEFT
   ,SPACE,DOWN,LEFT,SPACE}"                     <067>
33 PRINT"{HOME,3DOWN,10RIGHT,RVSON,3SPACE,
   DOWN,3LEFT,SPACE,DOWN,LEFT,2SPACE,DOWN,
   2LEFT,SPACE,DOWN,LEFT,SPACE}"                <194>
34 PRINT"{HOME,3DOWN,14RIGHT,RVSON,3SPACE,
   DOWN,3LEFT,SPACE,DOWN,LEFT,3SPACE,DOWN,
   3LEFT,SPACE,DOWN,LEFT,3SPACE}"               <051>
35 PRINT"{HOME,15DOWN,YELLOW}(C) 1983 BY C
   . NEUPERT{WHITE}"                            <096>
36 POKE 36874,T1:POKE 36875,T2:POKE 36876,
   T3:RETURN                                    <046>
40 PRINT"{HOME,21DOWN,WHITE,6RIGHT}HIT ANY
    KEY"                                        <253>
41 GET A$:IF A$=""THEN 41                       <050>
50 POKE 36878,15: PRINT"{CLR,3DOWN,WHITE}"
   :GE=0:PP=0                                   <029>
51 PRINT"{RIGHT,RVSON}1{RVOFF,SPACE}ENTERI
   NG OF":PRINT"{3RIGHT}COORDINATES"            <082>
52 PRINT:PRINT"{RIGHT,RVSON}2{RVOFF,SPACE}
   SINGLE STEP"                                 <187>
53 PRINT:PRINT"{RIGHT,RVSON}3{RVOFF,SPACE}
   AUTOMATIC MODE,"                             <035>
54 PRINT"{DOWN,RIGHT,RVSON}4{RVOFF,SPACE}M
   UTATIONS"                                    <053>
55 PRINT"{DOWN,RIGHT,RVSON}5{RVOFF,SPACE}M
   UTATIONS OFF"                                <069>
57 GET A$:IF A$=""THEN 57                       <227>
58 ON VAL(A$)GOTO 500,600,700,800,860           <146>
59 GOTO 57                                      <125>
99 STOP                                         <165>
500 POKE 36878,15:PRINT"{CLR,WHITE,DOWN}":
    PP=0:RD=0                                   <030>
517 PRINT"{2DOWN}ENTER NOW{SPACE,RVSON}S{R
    VOFF,2SPACE}FOR ":PRINT"{DOWN}SETTING
    DOTS, OR"                                   <203>
519 PRINT"{DOWN,RIGHT,RVSON}A{RVOFF,2SPACE
    }FOR DELETING ALL."                         <042>
520 REM                                         <074>
522 GET A$:IF A$=""THEN 522                     <144>
524 IF A$="S"THEN GOSUB 7000:GOTO 530           <227>
525 IF A$="A"THEN PRINT"{CLR,2DOWN,WHITE,S
    PACE}DELETING...":GOSUB 4000:GOTO 50        <200>
526 GOTO 522                                    <074>
530 PRINT"{CLR,DOWN}":GOSUB 9000:GOSUB 300
    0:FOR Y=1 TO 32:FOR X=1 TO 26:POKE 716
    8+X+27*Y,76                                 <209>
531 IF PEEK(5120+X+27*Y)=1 THEN POKE 7168+
    X+27*Y,42:POKE 7168+X+27*Y+30720,3:PP=
    PP+1                                        <232>
532 NEXT:NEXT:X=13:Y=16:AD=7613:POKE AD,16
    0:POKE AD+30720,2:POKE 36878,15             <218>
540 GET A$:IF A$=""THEN 540                     <034>
542 IF A$<>"" THEN Q=ASC(A$)                    <214>
544 IF Q=17 THEN Y=Y+1:GOTO 580                 <133>
545 IF Q=29 THEN X=X+1:GOTO 580                 <208>
546 IF Q=145 THEN Y=Y-1:GOTO 580                <090>
547 IF Q=157 THEN X=X-1:GOTO 580                <146>
548 IF Q=32 THEN 590                            <055>
549 IF Q=42 THEN 595                            <108>
550 IF Q=82 THEN PRINT"{CLR}":POKE 36879,7
    5:GOTO 5015                                 <131>
552 GOTO 540                                    <068>
580 IF Y>32 THEN Y=32                           <059>
581 IF Y<1 THEN Y=1                             <075>
582 IF X>26 THEN X=26                           <049>
583 IF X<1 THEN X=1                             <197>
584 POKE 36876,210::::::::::::::POKE 3687
    6,0:IF PEEK(AD)=42 THEN 587                 <128>
```

Listing 2. Hauptprogramm zu »LIFE«.
Bitte beachten Sie die Eingabehinweise auf Seite 76

```
585 IF PEEK(AD)=170 THEN POKE AD,42:POKE A
    D+30720,3:GOTO 587                   <096>
586 POKE AD,76:POKE AD+30720,1           <121>
587 AD=7168+X+27*Y:IF PEEK(AD)=42 THEN POK
    E AD,170:POKE AD+30720,2:GOTO 589    <002>
588 POKE AD,160:POKE AD+30720,2          <080>
589 GOTO 540                             <105>
590 POKE 36876,235:POKE AD,160:POKE AD+307
    20,2                                 <107>
592 POKE 5120+X+27*Y,0:POKE 36876,0:GOTO 5
    40                                   <009>
595 POKE 36876,240:POKE AD,170:POKE AD+307
    20,3                                 <208>
597 POKE 5120+X+27*Y,1:POKE 36876,0:PP=PP+
    1:GOTO 540                           <148>
599 RETURN                               <149>
600 GOSUB 2000                           <056>
610 PRINT"{CLR,3DOWN,WHITE}PRESS{SPACE,RVS
    ON}SPACE{RVOFF,SPACE}FOR EACH"       <022>
615 PRINT:PRINT"SINGLE STEP."            <216>
625 PRINT"{3DOWN,YELLOW,3SPACE}HIT ANY KEY
    {WHITE}"                             <085>
630 GET A$:IF A$=""THEN 630              <091>
635 PRINT"{CLR}":GOSUB 3000:GOSUB 9000   <229>
636 SYS 6400:GOSUB 11100:GOTO 641        <022>
640 SYS 6151:GOSUB 11000                 <014>
641 GET A$:IF A$=""THEN GOSUB 5000:GOTO 64
    1                                    <203>
642 IF ASC(A$)=136 THEN 5015             <187>
643 IF A$<>" "THEN 641                   <241>
650 GOTO 640                             <174>
700 GOSUB 2000                           <156>
702 PRINT"{CLR,DOWN}DO YOU WANT A HARDCOPY
    ":PRINT"{DOWN,2SPACE}[Y/N] ?"        <129>
703 GET A$:IF A$="" THEN 703             <133>
704 IF A$="N"THEN 707                    <132>
705 IF A$<>"Y"THEN 703                   <143>
706 PF=1:OPEN 1,4:CMD 1:PRINT#1          <027>
707 PRINT"{CLR}":GOSUB 3000:GOSUB 9000   <045>
710 SYS 6400:GOSUB 11100:FOR J=1 TO 900:NE
    XT:IF PF THEN C$="":ZX=0:ZY=0:GOSUB 17
    000                                  <252>
720 SYS 6151:GOSUB 11000:GOSUB 5000      <136>
721 C$="":ZX=0:ZY=0:IF PF=1 THEN GOSUB 170
    00                                   <139>
730 GET A$:IF A$=""THEN 720              <159>
740 IF ASC(A$)<>136 THEN 720             <115>
750 IF PF=1 THEN PRINT#1:CLOSE 1:PF=0    <005>
752 GOTO 5015                            <057>
800 PRINT"{CLR,3DOWN}RELATIVE FREQUENCY":P
    RINT"{2DOWN}FOR MUTATIONS?"          <167>
820 PRINT:PRINT:PRINT:INPUT FQ:IF FQ<0 OR
    FQ<>INT(FQ)THEN 820                  <108>
850 MF=1:GOTO 50                         <247>
860 PRINT"{CLR}": MF=0:FQ=0:FOR J=1 TO 500
    :NEXT:GOTO 50                        <232>
2000 PRINT"{CLR,WHITE,2DOWN}":RD=0       <044>
2002 PRINT"{DOWN,RVSON}F3{RVOFF,SPACE}SCRE
     EN COLOR":PRINT"{DOWN,RVSON}F5{RVOFF,
     SPACE}CHARACTER COLOR"              <135>
2003 PRINT"{DOWN,RVSON}F7{RVOFF,SPACE}GOTO
      MENUE"                             <082>
2005 PRINT"{3DOWN,WHITE}WANT RANDOM COLORS
      ?"                                 <230>
2010 GET A$:IF A$=""THEN 2010            <057>
2020 IF A$="Y"THEN RD=1                  <035>
2021 RETURN                              <045>
3000 POKE 648,28:SYS 58648:              <198>
3005 POKE 36864,8:POKE 36865,20:POKE 36866
     ,27:POKE 36867,194                  <018>
3010 POKE 36879,8:RETURN                 <154>
4000 FOR I=5120 TO 6018:POKE I,0:NEXT:RETU
     RN                                  <205>
5000 GET A$:IF A$=""THEN RETURN          <182>
5005 W=ASC(A$):IF W=134 THEN SC=SC+1+15*(S
     C=15):GOTO 5100                     <062>
5010 IF W=135 THEN 5020                  <188>
5012 IF W<>136 THEN A$="":RETURN         <069>
5015 POKE 36864,12:POKE 36865,38:POKE 3686
     6,150:POKE 36867,151:SYS 58648:POKE 3
     6879,75:GOTO 50                     <084>
5020 CC=CC+1+8*(CC=7)                    <008>
5025 FOR I=37888 TO 38900:POKE I,CC:NEXT:A
     $="":RETURN                         <168>
5100 POKE 36879,16*SC+8:A$="":RETURN     <038>
7000 PRINT"{CLR,DOWN}DO YOU WANT TO DELETE
     ":PRINT"{DOWN}THE OLD MATRIX [Y/N] ?" <042>
7010 GET A$:IF A$="J" THEN PRINT"{CLR}":GO
     SUB 4000:RETURN                     <018>
7020 IF A$="N" THEN RETURN               <204>
7030 GOTO 7010                           <192>
9000 FOR I=7660 TO 8185:POKE I,32:NEXT:FOR
     I=37888 TO 38800:POKE I,1+INT(RND(1)
     *8)*RD:NEXT:RETURN                  <171>
11000 IF MF=0 THEN 11100                 <088>
11010 FOR I=0 TO FQ:XM=INT(RND(1)*26)+1:YM
      =INT(RND(1)*32)+1                  <176>
11020 AM=5120+XM+27*YM:POKE AM,0         <225>
11030 NEXT                               <118>
11100 GE=GE+1:PP=PEEK(7160)+256*PEEK(7161)
      :PRINT"{HOME,WHITE,RIGHT}GEN.:";GE; <162>
11105 PRINT"{HOME,11RIGHT,RED}POPULATION{G
      REEN}";PP;"{LEFT,2SPACE,HOME}";    <145>
11110 IF PF THEN PRINT#1,"{4SPACE}GENERATI
      ON: ";GE;"{5SPACE}POPULATION: ";PP:P
      RINT#1                             <171>
11150 POKE 36878,15:POKE 36874,234:POKE 36
      875,223:POKE 36876,204:FOR J=1 TO 20
      0:NEXT                             <220>
11200 POKE 36874,0:POKE 36875,0:POKE 36876
      ,0:RETURN                          <030>
15000 FOR I=0 TO 8:READ D:POKE 6151+I,D:NE
      XT                                 <089>
16000 DATA 169,0,141,248,27,141,249,27,234 <199>
17000 FOR Y=1 TO 32:FOR X=1 TO 26:IF PEEK(
      5120+X+27*Y)=1 THEN C$=C$+"*":ZX=1:G
      OTO 17010                          <140>
17005 C$=C$+" "                          <118>
17010 NEXT:IF ZX THEN PRINT#1,"";TAB(25);C
      $:ZY=0:GOTO 17030                  <110>
17020 IF ZX=0 THEN ZY=ZY+1:IF ZY<4 THEN PR
      INT#1                              <235>
17030 C$="":ZX=0:NEXT Y:PRINT#1:PRINT#1:RE
      TURN                               <170>
```


Listing 2. Hauptprogramm zu »LIFE« (Schluß)



Herrscher über Leben und Tod

Es war schon immer der Reiz der Menschheit, Macht über andere auszuüben; erproben Sie Ihre Fähigkeit, über eine Stadt 65 Jahre zu regieren, lassen auch Sie sich von dieser Machtsucht erfassen und steigen Sie voll ein, in Hamurabi.

Das Programm »Hamurabi« für den C116/ C16 wurde dem berühmten Managementspiel »Kaiser« nachempfunden. Es geht darum, 65 Jahre lang über die Stadt Babylon zu herrschen.

Beispielsweise ist es wichtig, genügend Korn zu kaufen oder anzubauen, damit das Volk nicht verhungert...

Hamurabi läuft ohne Erweiterung. Der Spielstand kann auf Diskette gespeichert und natürlich wieder geladen werden. Wird die Frage »Altes Spiel laden?« mit JA beantwortet und es erscheint ein Disk-Error, muß überprüft werden, ob ein File mit dem alten Spielstand existiert. Erscheint keine Fehlermeldung, so wird das alte Spiel geladen. Gespeichert wird das Spiel, wenn am Ende einer Runde »S« für SAVEn eingegeben wird. Hierdurch wird ein File auf Diskette geschrieben, das alle wichtigen Daten enthält und das wieder geladen werden kann. Nach dem Programmstart wird das Titelbild aufgebaut und die Musik ertönt. Danach wird das Spiel mit »+« gestartet. Alle weiteren Anweisungen ergeben sich aus dem Programmablauf. Wir wünschen viel Spaß beim Spielen! (do)

Variablenliste	
ES	- Hektar Land, die eingesät werden sollen
BG	- Bargeld
EW	- Einwohner von Babylon
GM	- Anzahl der großen Mühlen
JA	- Jahr
KA	- Kornbesitz
KK	- Anzahl der Kornmühlen
LB	- Landbesitz
MP	- Anzahl der Marktplätze

Programmaufbau	
0 - 855	Vorspann
860 - 1100	Hauptmenü
2000 - 3100	Auslosung des Wetters
5000 - 8040	Kauf und Verkauf von Land und Korn
9000 - 9080	Kornverteilung
9100 - 9200	Bericht über die Einnahmen
9500 - 9800	Einsäen des Landes
9800 - 10000	Mühlen, etc. kaufen
10000 - 13000	Kommentar nach Ende des Spiels
15000 - 17000	Laden und Speichern des Spielstandes
17000 - 18000	Bericht über die Stadt (wird im Programmablauf angesprungen).

```
0 REM ------------------------------
1 REM ------------------------------
2 REM ----                      ----
3 REM ---- DOMINIK EISELE       ----
4 REM ----                      ----
5 REM ---- ALFRED-KUBIN-STR.13  ----
6 REM ----                      ----
7 REM ---- 5090 LEVERKUSEN 1    ----
8 REM ----                      ----
9 REM ---- WEST GERMANY         ----
10 REM ----                     ----
11 REM ---- TEL.0214/94695      ----
12 REM ----                     ----
13 REM -----------------------------
14 REM -----------------------------
15 REM ↑ REM-ZEILEN NICHT ABTIPPEN ↑
16 REM -----------------------------
17 PRINT"{CLR,BLACK,DOWN}";:COLOR0,11,6:
COLOR4,1,1:PRINTCHR$(8),CHR$(142)
18 PRINT"{2RIGHT,28SPACE}MUSIK VON{3SPAC
E}"
20 PRINT"{2RIGHT}£5£5£5{6SPACE}£5£5£5{10
SPACE}GERALD{6SPACE}"
21 PRINT"{2RIGHT}-ZXZX-{6SPACE}-ZXZX-{5S
PACE}L@{3SPACE}HANISCH{5SPACE}"
22 PRINT"{2RIGHT}JI{2SPACE}UK{6SPACE}JI{
2SPACE}UK{5SPACE}--{15SPACE}"
23 PRINT"{2RIGHT }- F-{8SPACE}-B -{6SPAC
E}--{15SPACE}"
24 PRINT"{2RIGHT }- V-£5£5£5£5-C -{4SPAC
E}NM--{6SPACE}NM{4SPACE}N{2SPACE}"
25 PRINT"{2RIGHT }-{2SPACE}ZXZXZXZXZX{2S
PACE}-M{2SPACE}N{2SPACE}M- NM{2SPACE}N{2
SPACE}M{2SPACE}N{3SPACE}"
26 PRINT"NM - F{10SPACE}B -- N{4SPACE}MN
{2SPACE}M -C - N{2SPACE}"
27 PRINT"{2RIGHT}M- V B{2SPACE}UI{2SPACE
}F C --N{10SPACE}M-{2SPACE}-N{5SPACE}"
28 PRINT"{2RIGHT }-{3SPACE}C{2SPACE}KJ{2
SPACE}V{3SPACE}@K{4SPACE}£5£5£5£5-C -£5{
4SPACE}"
29 PRINT"{2RIGHT }-{14SPACE}-{5SPACE}-ZX
ZXZXZX{2SPACE}ZX-{4SPACE}"
30 PRINT"{2RIGHT }-{6SPACE}UI{6SPACE}-{5
SPACE}- B{2SPACE}B{2SPACE}B{2SPACE}B -{4
SPACE}"
31 PRINT"{2RIGHT }-{4SPACE}UI--UI{4SPACE
}-{5SPACE}-{8SPACE}UI{2SPACE}-{4SPACE}"
32 PRINT"{2RIGHT }Z****EEEEEE****X{5SPAC
E}Z*********EE**X{4SPACE}"
97 PRINT"{DOWN}*DOMINIK EISELE* PRAESENT
IERT :"
99 PRINT"{RVSON,BROWN}+{38SPACE}+"
100 PRINT"{RVSON }+{2SPACE}+{2SPACE}++{2
SPACE}+++++ +{2SPACE}+ ++++{2SPACE}++{2S
PACE}++++ ++ "
101 PRINT"{RVSON }+{2SPACE}+ +{2SPACE}+
+ + + +{2SPACE}+ +{2SPACE}+ +{2SPACE}+ +
+ + ++ "
102 PRINT"{RVSON }++++ ++++ + + + +{2SPA
CE}+ ++++ ++++ ++++ ++ "
103 PRINT"{RVSON }+{2SPACE}+ +{2SPACE}+
+ + + +{2SPACE}+ + +{2SPACE}+{2SPACE}+ +
+ + ++ "
104 PRINT"{RVSON }+{2SPACE}+ +{2SPACE}+
+ + + ++++ +{2SPACE}+ +{2SPACE}+ ++++ ++
 ";
105 PRINT"{RVSON}+{38SPACE}+{RVOFF}";
106 RESTORE 200
107 VOL8
108 DO
109 READ X,Y
110 SOUND1,X,Y:VOL0:FORT=1TO5:NEXTT:VOL8
111 LOOP UNTIL X=0:GOTO 850
200 DATA 810,25,810,25,810,50,822,25,822
,25,822,50,870,25,870,25,870,25
201 DATA 870,12,810,112,798,25,798,25,79
8,50,784,25,784,25,784,50,770,25,770,25
202 DATA 770,25,770,12,770,37,810,25,834
,18,854,6,864,18,881,6,897,25,897,12
203 DATA 897,63,881,25,881,12,881,63,870
,25,870,25,870,25,870,12,810,112
```

```
204 DATA 864,25,864,25,881,25,881,25,897
,25,897,12,870,38,870,25,897,11,897,11
205 DATA 897,11,897,11,881,25,881,25,864
,25,810,12,864,62,1,1,0,0
850 GETR$:IFR$="+" THEN VOL8:SOUND1,810,
20:GOTO 860
855 GOTO 850
860 KA=15000:AA=0:BG=23000:EW=INT(257*RN
D(1))+500:LB=10000:JA=1:MP=0:KK=0:GM=0
870 KR=INT(10000*RND(1))+60000:BJ=0:MP=1
:BB=0
871 PRINT"{CLR,4DOWN,3RIGHT}ALTES SPIEL
LADEN ?"
872 PRINT"{DOWN,5RIGHT}(J/N)"
873 GETD$:IFD$="J"THENVOL8:GOTO 15000
876 IFD$="N" THEN VOL8:SOUND1,810,20:GOT
O890
877 GOTO 873
890 GOSUB 1000
900 GOTO 890
1000 COLOR 4,1,7:COLOR0,1,1:PRINT"{CLR}"
1004 GOSUB 3000
1005 GOSUB 2000
1006 PRINT"{CLR,RED}"KR"{YELLOW}SCHEFFEL
 KORN WERDEN ANGEBOTEN{RED}"
1007 PRINT"{YELLOW,RIGHT}ERNTEERTRAEGE :
"INT(EE)"SCHEFFEL"
1008 PRINTVK"{YELLOW}% EURER KORNRESERVE
N SIND VERFAULT"
1009 GOSUB 3000
1010 BK=EW*9
1011 PRINT"{BLUE}************************
*****************";
1012 PRINT"{RIGHT,YELLOW}KORNRESERVEN (S
CHEFFEL) ---{SPACE,GREY 3}";INT(KA)
1020 PRINT"{RIGHT,DOWN,YELLOW}BENOETIGTE
S KORN{8SPACE}---{SPACE,GREY 3}";INT(BK)
1030 PRINT"{RIGHT,DOWN,YELLOW}KORNPREIS
(TALER){7SPACE}---{SPACE,GREY 3}";INT(KP
)
1040 PRINT"{RIGHT,DOWN,YELLOW}LANDPREIS
(TALER){7SPACE}---{SPACE,GREY 3}";INT(LP
)
1050 PRINT"{RIGHT,DOWN,YELLOW}LANDBESITZ
 (HEKTAR){5SPACE}---{SPACE,GREY 3}";INT(
LB)
1060 PRINT"{RIGHT,DOWN,YELLOW}BARGELD (T
ALER){9SPACE}---{SPACE,GREY 3}";INT(BG)
1070 PRINT"{BLUE}************************
*****************{LIG.GREEN}";
1081 PRINT"1 ----- KORN KAUFEN{4SPACE}--
---{2SPACE}QYYYYYYYYY";
1082 PRINT"2 ----- KORN VERKAUFEN -----{
2SPACE}HWEITER{2SPACE}"
1083 PRINT"3 ----- LAND KAUFEN{4SPACE}--
---{2SPACE}HMIT"CHR$(130)"{SPACE,RVSON,G
REY 1}SPACE{LIG.GREEN,RVOFF}"
1084 PRINT"4 ----- LAND VERKAUFEN -----{
2SPACE}LPPPPPPPPP";
1088 GET A$:IFA$=" " THEN VOL8:SOUND1,90
0,10:GOTO 9000
1089 IFA$="1" THEN VOL8:SOUND1,100,10:GO
TO 5000
1090 IFA$="2" THEN VOL8:SOUND1,150,10:GO
TO 6000
1091 IFA$="3" THEN VOL8:SOUND1,250,10:GO
TO 7000
1092 IFA$="4" THEN VOL8:SOUND1,340,10:GO
TO 8000
1093 GOTO 1088
1100 RETURN
2000 REM --- KORNRESERVEN ---
2010 K1=KA/100
2015 VK=INT(100*RND(1))
2020 KO=K1*VK
2030 KA=KA-KO
2035 IF KA<=0 THEN KA=0
2040 RETURN
3000 IF BB=1 THEN PRINTW$:RETURN
3005 KP=INT(370*RND(1))+119
3010 IF KP<192 THEN W$=" TOLLES WETTER{6
SPACE}SEHR REICHE ERNTE":HL=2
3020 IF KP>=192 AND KP<284THENW$=" GUTES
 WETTER{8SPACE}GEWOEHNLICHE ERNTE":HL=3
3030 IF KP>=284 AND KP<380 THEN W$=" REG
EN{12SPACE}SCHLECHTE ERNTE":HL=4
3040 IF KP>=380 THEN W$=" DUERRE{9SPACE}
HUNGERSNOT DROHT":HL=5
3050 LP=INT(70*RND(1))+19
3100 BB=1:RETURN
5000 INPUT"{DOWN}WIEVIEL KORN WOLLT IHR
KAUFEN";KM
5001 IFAA=1ANDKR<=0THENPRINT"{RVSON}IHR
BEKOMMT NUN KEIN KORN MEHR !!!!!!{RVOFF}
":GOTO 5010
5003 IF KR<=0 THEN KR=0
5005 IF KM<0 THEN 1006
5006 IFKM>KRTHENGOTO 5050
5007 KA=KA+KM:KR=KR-KM:BG=BG-KM*KP/1000
5010 VOL8:SOUND1,900,15:FORT=1TO500:NEXT
T
5020 GOTO 1006
5050 PRINT"{RVSON}SO VIEL KORN WIRD NICH
T ANGEBOTEN !{RVOFF}":AA=1:GOTO5010
6000 INPUT"{DOWN}WIEVIEL KORN WOLLT IHR
VERKAUFEN";KV
6010 IF KV>KA THEN PRINT"{RVSON}IHR HABT
 NICHT SO VIEL KORN !!!{RVOFF}":GOTO 501
0
6020 IF KV<0 THEN 1006
6035 KR=KR+KV:KA=KA-KV:BG=BG+KV*KP/1000
6040 GOTO 5010
7000 INPUT"{DOWN}WIEVIEL LAND WOLLT IHR
KAUFEN";LW
7010 BG=BG-LW*LP/10
7020 LB=LB+LW
7025 IF LW<0 THEN 1006
7030 VOL8:SOUND1,900,15:FORT=1TO500:NEXT
T
7040 RETURN
8000 INPUT"{DOWN}WIEVIEL LAND WOLLT IHR
VERKAUFEN";LV
8005 IF LV>LB THEN PRINT"{RVSON}SO VIEL
LAND HABT IHR NICHT !!!{RVOFF}":GOTO 703
0
8010 BG=BG+LP*LV/10
8020 IF LV<0 THEN 1006
8030 LB=LB-LV
8040 GOTO 7030
9000 PRINT"{CLR,WHITE}":COLOR 0,1,1:COLO
R4,11,1
9005 PRINT"{2DOWN,13RIGHT}KORNVERTEILUNG
"
9010 PRINT"{4RIGHT,2DOWN}O HAMURABI VON
BABYLON ICH FRAGE"
9020 PRINT"{4RIGHT,DOWN}EUCH, WIEVIEL KO
RN WOLLT IHR DEN"
9022 PRINT"{4RIGHT,DOWN}HUNGERNDEN ZUR V
ERFUEGUNG STELLEN?"
9025 PRINT"{DOWN,4RIGHT}IHR BESIZT{RED}"
INT(KA)"{WHITE}SCHEFFEL KORN"
9026 PRINT"{DOWN,4RIGHT}DIE BENOETIGTE K
ORNMENGE IST{RED}"INT(BK)
9027 PRINT"{DOWN,4RIGHT,WHITE}DIE EINWOH
NERZAHL BETRAEGT{RED}"EW
```

Listing »Hamurabi«

```
9030 INPUT"{WHITE,2DOWN,4RIGHT}BETRAG{SP
ACE,GREEN}";VS
9032 IF VS>KA THEN VOL8:SOUND1,150,15:GO
TO9000
9033 IF VS<0 THEN VOL8:SOUND1,150,15:GOT
O 9000
9035 KA=KA-VS
9040 VR=INT(VS/EW):TW=EW
9050 VOL8:SOUND1,350,15:GOSUB 9500
9055 PRINT"{12RIGHT,3DOWN}WEITER MIT"CHR
$(130)"{SPACE,RVSON,BLUE}SPACE{RVOFF}"
9070 GETS$:IFS$=" " THEN VOL8:SOUND1,900
,15:GOTO 9100
9080 GOTO 9070
9100 PRINT"{CLR,WHITE}":COLOR0,11,4:COLO
R4,11,6
9101 PRINT"{DOWN,8RIGHT}O HAMURABI VON B
ABYLON"
9102 PRINT"{DOWN,2RIGHT}ICH BERICHTE EUC
H UEBER EURE STADT..{4SPACE}"
9103 PRINT"{DOWN,17RIGHT}JAHR"JA
9105 PRINT"{DOWN,RIGHT,WHITE}"INT(TW)"{S
PACE,BLACK}EINWOHNER SIND VERHUNGERT."
9110 IFSS=1 THEN PRINT"{RIGHT,WHITE}"ZW"
{BLACK}EINWOHNER SIND ZUGEWANDERT"
9120 AG=INT(47*RND(1))+20
9130 PRINT"{RIGHT,YELLOW}"INT(AG)"{SPACE
,BLACK}EINWOHNER WURDEN GEBOREN."
9140 AT=INT(47*RND(1))+20
9150 PRINT"{RIGHT,YELLOW}"INT(AT)"{SPACE
,BLACK}EINWOHNER SIND IM ALTER GESTORBEN
":EW=EW+AG-AT+ZW-TW
9151 PRINT"{DOWN,RIGHT,YELLOW}EURE MARKT
PLAETZE BRACHTEN"INT(MP*495):BG=BG+INT(M
P*495)
9152 PRINT"{DOWN,RIGHT,YELLOW}EURE KORNM
UEHLEN BRACHTEN"INT(KK*589):BG=BG+INT(KK
*589)
9153 PRINT"{DOWN,RIGHT,YELLOW}EURE GR. K
ORNMUEHLEN BRACHTEN"INT(GM*678):BG=BG+(G
M*678)
9154 PRINT"{DOWN,RIGHT,BLACK}(ANGABEN IN
 TALERN){YELLOW}"
9155 SV=SV+EW*2:IF SV<0 THEN SV=0
9156 PRINT"{DOWN,RIGHT,WHITE}DAS VOLK ZA
HLTE"SV"TALER STEUERN."
9157 BG=BG+SV
9158 SV=0
9160 PRINT"{DOWN,BLACK,12RIGHT}WEITER MI
T"CHR$(130)"{SPACE,RVSON}SPACE{RVOFF}"
9170 GETC$:IFC$=" " THEN VOL8:SOUND1,900
,15:GOTO 17000
9180 GOTO 9170
9200 GOTO 10000
9500 FOR R=1 TO 8
9510 IF VR<=R THEN EW=INT(EW-(TW/(R+R)))
:TW=TW-EW:RETURN
9520 NEXTR
9530 IF VR=>9 ANDVR<=25THEN TW=0:RETURN
9540 FOR R=26 TO 20000
9550 IFR=VRTHEN ZW=INT((TW*(R/2))/98):EW
=INT(EW+(TW*(R/2))/98):TW=0:SS=1:RETURN
9560 NEXTR
9600 RETURN
9700 PRINT"{CLR,WHITE}":COLOR 0,1,1:COLO
R 4,11,2
9705 PRINT"{2DOWN,15RIGHT}EINSAEHEN"
9710 PRINT"{2DOWN,2RIGHT}WEITERHIN O HAM
URABI FRAGE ICH EUCH :"
9720 PRINT"{DOWN,2RIGHT}WIEVIEL LAND WOL
LT IHR EINSAEEN ?"
9730 PRINT"{DOWN,2RIGHT}IHR BESIZT{RED}"
LB"{WHITE}HEKTAR LAND."
9740 PRINT"{DOWN,2RIGHT}IHR HABT{RED}"IN
T(BG)"{WHITE}TALER."
9745 PRINT"{DOWN,2RIGHT}DAS EINSAEEN PRO
 HEKTAR LAND"
9746 PRINT"{DOWN,2RIGHT}KOSTET{RED}"HL"{
WHITE}TALER."
9750 INPUT"{DOWN,2RIGHT}(HEKTAR) ";ES
9760 IF ES>LB THEN VOL8:SOUND1,150,15:GO
TO9700
9770 IF ES<0 THEN VOL8:SOUND1,150,15:GOT
O9700
9775 ZF=INT(24*RND(1))+72
9780 BG=INT(BG-HL*ES):KA=KA+(ES/HL)*ZF:E
E=INT((ES/HL)*ZF):VOL8:SOUND1,350,15
9785 PRINT"{2DOWN,12RIGHT}WEITER MIT"CHR
$(130)"{SPACE,RVSON}SPACE{RVOFF}"
9795 GETR$:IFR$=" " THEN VOL8:SOUND1,900
,15:GOTO 9800
9796 GOTO 9795
9800 PRINT"{CLR,BLACK}":COLOR 0,12,4:COL
OR4,1,4
9810 PRINT"{4RIGHT}O EDLER HAMURABI,"
9820 PRINT"{3RIGHT,2DOWN}UM DIE STAATSKA
SSE AUFZUBESSERN,"
9830 PRINT"{3RIGHT,DOWN}WAERE ES VON VOR
TEIL EINNAHMEQUELLEN"
9840 PRINT"{3RIGHT,DOWN}ZU ERRICHTEN."
9850 PRINT"{3RIGHT,DOWN}EURE WAHL :"
9860 IFLB/(MP+KK+GM)<1000THENPRINT"{2DOW
N,RIGHT,RVSON}IHR BENOETIGT MEHR BAULAND
!{RVOFF}":GOTO 9990
9959 PRINT"{2DOWN}TALER"BG
9960 INPUT"WIEVIEL MARKTPLAETZE - 2000 T
ALER";TM
9961 MP=MP+TM
9962 BG=BG-TM*2000
9965 VOL8:SOUND1,180,15
9969 PRINT"TALER"BG
9970 INPUT"WIEVIEL KORNMUEHLEN{2SPACE}-
3000 TALER";TK
9971 KK=KK+TK
9972 BG=BG-TK*3000
9975 VOL8:SOUND1,250,15
9979 PRINT"TALER"BG
9980 INPUT"WIEVIEL GROSSE MUEHLEN - 5000
 TALER";TG
9981 GM=GM+TG
9982 BG=BG-TG*5000
9985 VOL8:SOUND1,300,15
9986 PRINT"TALER"BG
9990 KR=INT(10000*RND(1))+60000
9991 IF BG<0 THEN BJ=BJ+BG
9992 IF BJ<-90000 THEN 10000
9993 IF EW<500 THEN 10000
9994 IF JA=65 THEN 12000
9995 PRINT"{DOWN,12RIGHT}S=SAVEN!{RVOFF}
"
9996 PRINT"{DOWN,12RIGHT}WEITER MIT"CHR$
(130)"{SPACE,RVSON}SPACE{RVOFF}"
9997 IFR$="S" THEN BB=0:AA=0:SS=0:JA=JA+
1:GOSUB 16000
9998 GETR$:IFR$=" " THEN BB=0:AA=0:SS=0:
JA=JA+1:VOL8:SOUND1,900,15:GOTO1000
9999 GOTO 9997
10000 FORT=1TO2000:NEXTT:PRINT"{CLR,WHIT
E}":COLOR0,1,1:COLOR4,1,1
10005 PRINT"{2DOWN,RIGHT}HAMURABI ICH MU
SS EUCH SAGEN,DASS IHR"
10010 PRINT"{DOWN,RIGHT}EIN SCHLECHTER H
ERRSCHER WART."
10020 PRINT"{DOWN,RIGHT}EUER VOLK ZAEHLT
E NACH"JA"JAHREN"
10030 PRINT"{DOWN}"EW"MENSCHEN."
```

```
10040 PRINT"{DOWN,RIGHT}EUER LAND WAR"LB
"HEKTAR GROSS UND"
10050 PRINT"{DOWN,RIGHT}EURE SCHATZKAMME
R WAR MIT"
10060 PRINT"{DOWN,RIGHT}"BG"TALERN GEFUE
LLT."
10070 PRINT"{DOWN,RIGHT}DOCH SCHAETZT EU
RE FAEHIGKEITEN SELBST"
10080 PRINT"{DOWN,RIGHT}EIN UND UEBERLEG
T OB FUER EUCH EINE"
10090 PRINT"{DOWN,RIGHT}WIEDERWAHL IN FR
AGE KOMMT....."
10095 GOSUB10500
10100 PRINT"{DOWN,RIGHT}({RVSON}J{RVOFF}
A/{RVSON}N{RVOFF}EIN)"
10110 GETF$:IFF$="J" THEN VOL8:SOUND1,70
0,20:RUN
10120 IFF$="N" THEN VOL8:SOUND1,200,20:P
RINT"{CLR}":GOTO 13000
10130 GOTO10110
10500 RESTORE 10510
10505 VOL8
10510 DO
10515 READ X,Y
10516 SOUND1,X,Y:VOL0:FORT=1TO5:NEXTT:VO
L8
10517 LOOP UNTIL X=0:RETURN
10518 DATA 453,60,453,45,453,15,453,60
10519 DATA 488,45,345,15,453,45,345,15,4
53,45,345,15,453,120,1,1,0,0
12000 FORT=1TO2000:NEXTT:PRINT"{CLR,WHIT
E}":COLOR0,1,1:COLOR4,1,1
12010 PRINT"{2DOWN,RIGHT}HAMURABI VON BA
BYLON ICH SAGE EUCH"
12020 PRINT"{DOWN,RIGHT}NACHDEM IHR 65 J
AHRE LANG DER STADT"
12030 PRINT"{DOWN,RIGHT}VORGESTANDEN HAB
T WIRD ES NUN"
12040 PRINT"{DOWN,RIGHT}SCHWER SEIN EINE
N EBENBUERTIGEN"
12050 PRINT"{DOWN,RIGHT}NACHFOLGER ZU FI
NDEN."
12060 PRINT"{DOWN,RIGHT}IHR WART SEIT JA
HRHUNDERTEN DER BESTE"
12065 PRINT"{DOWN,RIGHT}HERRSCHER UEBER
B A B Y L O N !!!!!"
12070 PRINT"{2DOWN,2RIGHT}LANDBESITZ :"L
B"HEKTAR."
12080 PRINT"{2RIGHT}BARGELD{4SPACE}:"BG"
TALER."
12090 PRINT"{2RIGHT}EINWOHNER{2SPACE}:"E
W
12091 GOSUB12500
12095 PRINT"{2DOWN,2RIGHT,RVSON,RED}SPAC
E{RVOFF,WHITE }DRUECKEN..."
12100 GETAA$:IFAA$=" "THEN PRINT"{CLR}":
SOUND1,300,20:GOTO13000
12110 GOTO12100
12500 RESTORE12520:VOL8
12510 DO
12515 READ X,Y
12516 SOUND1,X,Y:VOL0:FORT=1TO4:NEXTT:VO
L8
12517 LOOP UNTIL X=0:RETURN
12520 DATA 596,20,596,20,704,20,739,20,7
70,40,704,10,685,10,643,20,784,20
12521 DATA 784,20,784,40,770,10,784,10,8
10,30,704,10,704,20,704,20,685,20
12522 DATA 704,20,739,100,596,10,596,10,
596,20,704,20,739,20,770,40,704,10
12523 DATA 685,10,643,20,784,20,784,20,7
84,40,784,10,784,10,770,30,739,10,704,20
12524 DATA 596,20,685,20,739,20,704,100,
810,20,834,20,834,20,834,20,834,20
12525 DATA 854,30,864,10,810,20,810,20,8
10,20,770,40,810,10,810,10,864,30
12526 DATA 810,10,810,20,810,20,784,20,7
70,20,739,100,810,10,810,10,834,20
12527 DATA 834,20,834,20,834,20,854,20,8
64,20,810,30,810,10,810,20,770,40
12528 DATA 810,10,810,10,864,30,810,10,8
10,20,810,20,784,20,770,20,810,100
12529 DATA 1,1,0,0
12590 END
13000 PRINT"{8DOWN,14RIGHT,RVSON,GREEN}!
SPIELENDE!{RVOFF}":FORT=1TO60:NEXTT:GOTO
13000
14000 END
15000 REM --- LADEN ---
15010 OPEN 2,8,2,"0:HAM.SEQ.,S,R":OPEN15
,8,15:INPUT#15,EN,EM$,ET,ES
15020 INPUT#2,AA,BB,BG,EE,EW,GM
15030 INPUT#2,JA,KA,KK,LB,MP,SS
15040 PRINT"{3DOWN,3RIGHT}"
15041 IF EM$="OK"THENPRINT"OK":CLOSE2,8,
2:CLOSE15:FORT=1TO1000:NEXTT:EM$="":GOTO
 890
15042 IFEM$<>"OK"THENPRINT"DISK ERROR":C
LOSE2:CLOSE15:FORT=1TO1000:NEXT:GOTO871
16000 REM --- SAVEN ---
16005 OPEN 2,8,2,"@:HAM.SEQ.,S,W"
16010 PRINT#2,AA:PRINT#2,BB:PRINT#2,BG:P
RINT#2,EE:PRINT#2,EW:PRINT#2,GM
16020 PRINT#2,JA:PRINT#2,KA:PRINT#2,KK:P
RINT#2,LB:PRINT#2,MP:PRINT#2,SS
16030 CLOSE2,8,2:PRINT"{HOME,5RIGHT,22DO
WN}"DS$:RETURN
17000 PRINT"{CLR,WHITE}":COLOR 0,1,1:COL
OR4,1,1
17005 VOL8:SOUND1,400,60:FORT=1TO5:NEXTT
:SOUND1,500,70
17010 PRINT"{3DOWN}WEITERHIN EREIGNETE S
ICH IN DIESEM JAHR:"
17022 PP=INT(8*RND(1))+1
17030 IF PP=1THEN PRINT"{2DOWN}EURE FEIN
DE NAHMEN EUCH BEI EINEM":LB=LB-1000
17031 IFPP=1THENPRINT"{2DOWN}RAUBZUG 100
0 HEKTAR DES LANDES.":GOTO17090
17040 IFPP=2THENPRINT"{2DOWN}EINE HUNGER
SNOT BEDROHTE EURE BUERGER":EW=EW-50
17041 IFPP=2THENPRINT"{2DOWN}UND NAHM EU
CH WEITERE 50 EINWOHNER.":GOTO17090
17050 IFPP=3THENPRINT"{2DOWN}DIE PEST ZO
G UEBER DIE STADT UND NAHM":EW=EW-40
17051 IFPP=3THENPRINT"{2DOWN}EUCH 40 STE
UERZAHLER.":GOTO 17090
17060 IFPP=4THENPRINT"{2DOWN}EIN SANDSTU
RM VERWUESTETE DIE STADT.":BG=BG-5000
17061 IFPP=4THENPRINT"{2DOWN}FUER EINE S
AEUBERUNG ZAHLT 5000 TALER.":GOTO 17090
17070 IFPP=5THENPRINT"{2DOWN}IHR HABT DU
RCH EINE GEWONNENE SCHLACHT":LB=LB+1500
17071 IFPP=5THENPRINT"{2DOWN}1500 HEKTAR
 LAND DAZU BEKOMMEN.":GOTO 17090
17080 IFPP>5THENPRINT"{2DOWN}IHR HABT GL
UECK,ES GESCHAH NICHTS"
17081 IFPP>5THENPRINT"{2DOWN}BESONDERES.
":GOTO 17090
17090 PRINT"{4DOWN,RED,12RIGHT}WEITER MI
T"CHR$(130)"{SPACE,RVSON}SPACE{RVOFF}"
17091 GETC$:IFC$=" " THEN VOL8:SOUND1,90
0,15:GOTO 9700
17092 GOTO 17091
18000 END
```

©64'er Listing »Hamurabi« (Schluß)

Rate oder hänge!

Haben Sie Spaß an Ratespielen? Dann wird Ihnen »Galgenraten« gefallen. Aber es geht um Kopf und Kragen.

Der Computer im Haus erspart den Zimmermann. Das klingt zwar etwas wunderlich, aber wenn Sie das Spiel eingetippt und gestartet haben, wird es sich nur selten verhindern lassen, daß Sie in der schon genannten Holzbaubranche tätig werden – zumindest auf dem Bildschirm. Eigentlich geht es in dem Spiel ja darum, ein bestimmtes Wort zu erraten, aber mit jedem falsch geratenen Buchstaben forcieren Sie den Bau einer Holzkonstruktion, die in ihrer nüchternen Zweckform auch »Galgen« genannt wird. Und wenn Sie Ihrem Ebenbild den Verbrechertod nach alter englischer Tradition ersparen wollen, dann sollten Sie möglichst bald auf das richtige Wort kommen.

Aber jetzt wollen wir uns den Details zuwenden. Das Spiel besteht aus zwei Programmteilen, wobei das Vorprogramm »Galgenraten« (Listing 1) zuerst geladen werden muß. In diesem Programmteil wird der Zeichensatz neu definiert. Nach dem Laden des »Hauptprogramms« (Listing 2) können Sie wählen, ob der Computer oder ein Mitspieler Ihnen das zu erratende Wort vorgeben soll. Anschließend wird Ihnen durch Striche, auch Platzhalter genannt, das zu erratende Wort auf dem Bildschirm symbolisiert. Wenn Sie nun einen Buchstaben über die Tastatur eingeben, erscheint dieser an entsprechender Stelle des Wortes – vorausgesetzt, er ist im Wort enthalten. War der Buchstabe falsch, beginnt der schon erwähnte Bau. Wenn Sie durch ungünstige Buchstabenkombination die Hänkerstat vollendet haben, können Sie das Spiel nach einer Trauermelodie mit einem neuen Wort fortsetzen. Selbstverständlich werden Sie nicht daran gehindert, nach richtigem Raten das Spiel ebenfalls fortzusetzen. Die Abschlußmelodie hat in diesem Fall auch nicht mehr den traurigen Charakter.

Zum Schluß noch ein Tip, den Sie beim Eintippen des »Hauptprogramms« (Listing 2) beachten sollten. Die Zeilennummern der Datenfelder dürfen Sie nicht verändern, da im Programmverlauf auf diese direkt zugegriffen wird.

(Swen Jung/kn)

```
0 COLOR0,1:COLOR4,1
1 PRINT"{CLR,YELLOW}"
2 POKE51,0:POKE52,60:POKE55,0:
POKE56,60
3 FORA=15360TO16191
4 READA$
5 POKEA,DEC(A$)
6 NEXT
7 POKE65298,8:POKE65299,60
8 CHAR1,4,11,"[illegible]"
9 NEW
10 DATA00,00,00,00,00,00,7E,00
11 DATA18,24,24,7E,62,62,62,00
12 DATA78,24,24,3C,32,32,7C,00
13 DATA3C,42,40,60,60,62,3C,00
14 DATA7C,22,22,32,32,32,7C,00
15 DATA7E,22,20,38,20,22,7E,00
16 DATA7E,22,20,38,20,20,70,00
17 DATA3C,42,40,6E,6A,62,3C,00
18 DATA42,42,42,7E,62,62,62,00
19 DATA3C,10,10,18,18,18,3C,00
20 DATA0E,04,04,0C,0C,4C,38,00
21 DATA44,48,50,7C,62,62,62,00
22 DATA40,40,40,60,60,62,7E,00
23 DATA24,5A,42,62,62,62,62,00
24 DATA32,4A,4A,6A,6A,6A,64,00
25 DATA3C,42,42,62,62,62,3C,00
26 DATA7C,22,22,3C,30,30,30,00
27 DATA3C,42,42,62,62,6A,3C,06
28 DATA78,24,24,38,34,32,32,00
29 DATA3C,42,40,3C,06,46,3C,00
30 DATA7E,08,08,18,18,18,18,00
31 DATA42,42,42,62,62,62,3C,00
32 DATA42,42,42,34,34,34,18,00
33 DATA46,46,46,46,42,5A,24,00
34 DATA42,42,24,18,34,62,62,00
35 DATA42,42,24,18,18,18,18,00
36 DATA7E,42,04,18,30,62,7E,00
37 DATA00,18,3C,24,24,24,24,7E
38 DATA7E,62,62,62,62,62,62,00
39 DATA00,7E,7E,22,22,20,20,3C
40 DATA3C,30,30,32,32,7E,7E,00
41 DATA00,3C,7E,42,42,40,40,6E
42 DATA00,00,00,00,00,00,00,00
43 DATA06,06,06,06,00,06,06,00
44 DATA6E,6A,6A,62,62,7E,3C,00
45 DATA00,40,40,40,40,40,40,60
46 DATA60,60,60,62,62,7E,7E,00
47 DATA00,32,7A,4A,4A,4A,4A,6A
48 DATA6A,6A,6A,6A,6A,6E,64,00
49 DATA00,78,7C,44,44,44,44,78
50 DATA78,64,62,62,62,62,62,00
51 DATA00,7E,10,10,10,10,10,18
52 DATA18,18,18,18,18,18,18,00
53 DATA01,03,07,0F,1F,3F,7F,FF
54 DATA00,00,00,00,00,06,06,0C
55 DATAFF,FF,FF,FF,FF,FF,FF,FF
56 DATA00,00,00,00,00,06,06,00
57 DATA80,C0,E0,F0,F8,FC,FE,FF
58 DATA3C,42,46,6A,72,62,3C,00
59 DATA08,18,08,18,18,18,7E,00
60 DATA3C,42,02,0C,30,[illegible],7E,00
61 DATA3C,42,02,1C,06,46,3C,00
62 DATA40,40,48,48,7E,18,18,00
63 DATA7E,40,7C,06,06,46,3C,00
64 DATA3C,42,40,7C,46,46,3C,00
65 DATA7E,42,04,18,18,18,18,00
66 DATA3C,42,42,3C,46,46,3C,00
67 DATA3C,42,42,3E,06,46,3C,00
68 DATA00,06,06,00,06,06,00,00
69 DATAE0,70,38,1C,0E,07,03,01
70 DATA01,01,01,01,01,01,01,01
71 DATA80,80,80,80,80,80,80,80
72 DATA07,1F,3F,78,70,E4,E0,E0
73 DATA3C,42,02,0C,18,00,18,00
74 DATAE0,F8,FC,9E,0E,27,07,07
75 DATAE1,C0,80,43,44,20,1C,07
76 DATA87,03,01,C2,22,04,38,E0
77 DATA07,02,1C,78,FC,FE,FF,FF
78 DATAE0,40,38,1E,3F,7F,FF,FF
79 DATA7F,FF,FF,FF,7F,FF,FF,FF
80 DATA00,00,00,00,01,07,1F,3F
81 DATA00,00,00,00,80,E0,F8,FC
82 DATA3F,7F,7F,7F,7E,7E,7E,7E
83 DATAFC,FE,FE,FE,7E,7E,7E,7E
84 DATA7E,7E,7E,7E,7E,7E,7E,7E
85 DATA7E,24,24,42,42,24,18,00
86 DATA80,FF,FF,FF,FF,FF,FF,FF
87 DATA01,FF,FF,FF,FF,FF,FF,FF
88 DATAFE,FE,FE,FE,FE,FE,FE,FE
89 DATA7F,7F,7F,7F,7F,7F,7F,7F
90 DATA24,66,7E,7E,7E,7E,3C,18
91 DATA00,00,38,04,3C,4C,3C,00
92 DATA40,40,78,64,64,64,78,00
93 DATA00,00,3C,60,60,60,3C,00
94 DATA04,04,3C,64,64,64,3C,00
95 DATA00,00,38,64,7C,60,3C,00
96 DATA00,00,3C,64,64,3C,04,78
97 DATA40,40,78,64,64,64,64,00
98 DATA00,20,00,30,30,30,78,00
99 DATA40,40,48,50,70,68,64,00
100 DATA30,10,10,30,30,30,78,00
101 DATA00,00,34,6A,6A,6A,6A,00
102 DATA00,00,38,64,64,64,64,00
103 DATA00,00,38,64,64,64,38,00
104 DATA00,00,58,64,64,58,40,40
105 DATA00,00,38,64,60,60,60,00
106 DATA00,00,38,40,38,0C,78,00
107 DATA10,7C,10,18,18,18,18,00
108 DATA00,00,64,64,64,64,38,00
109 DATA00,00,62,62,6A,6A,34,00
110 DATA00,00,64,64,64,3C,04,78
111 DATA00,00,7C,04,38,60,7C,00
112 DATA28,00,38,64,64,64,38,00
113 DATA28,00,64,64,64,64,38,00
```

Listing 1. »Galgenraten« – Vorprogramm

```
0 DATASCHLITTSCHUH
1 DATAREGENWURM
2 DATAPHYSIKBUCH
3 DATAHAUSMANNSKOST
4 DATAOBSERVATORIUM
5 DATAMASKOTTCHEN
6 DATAKNOCHENBRUCH
7 DATAFERIENJOB
8 DATATAUCHERBRILLE
9 DATADARMVERSTOPFUNG
10 DATAHAREMSDAME
11 DATAAUTOWASCHANLAGE
12 DATADIRNDLKLEID
13 DATATRACHTENANZUG
14 DATATIEFSCHLAG
15 DATAKANNIBALISMUS
16 DATAQUIZSENDUNG
17 DATAPLASTIKBEUTEL
18 DATAPORTEMONAI
19 DATAPREISELBEERE
20 DATAGROSCHENROMAN
21 DATATASCHENRECHNER
22 DATAKARNICKELSTALL
23 DATACOMPUTERFREAK
24 DATAARZNEIMITTEL
25 DATAMEDIKAMENT
26 DATAJONGLEUR
27 DATAINSTALLATEUR
28 DATACHAMPIGNON
29 DATAINFORMATION
30 DATATISCHTENNIS
31 DATARADIORECORDER
32 DATALIEBESPAAR
33 DATAANTIALKOHOLIKER
34 DATABUNDESKANZLER
```

Listing 2. Hauptprogramm zu Galgenraten

```
35 DATABABYSCHNULLER
36 DATASPIEGELBILD
37 DATACELLOPHAN
38 DATAWASSERKLOSETT
39 DATAZOOLOGIE
40 DATAHAIFISCHFLOSSE
41 DATAVOKABELHEFT
42 DATAFREUNDSCHAFT
43 DATAGESELLSCHAFT
44 DATAHAARWUCHSMITTEL
45 DATAKIRCHTURMUHR
46 DATAHONIGMELONE
47 DATAGASTHAUS
48 DATADURCHSCHNITT
49 DATASCHNURRBART
50 DATAKINDERGARTEN
51 DATAORIENTTEPPICH
52 DATATEMPERAMENT
53 DATASTUHLGANG
54 DATAMIEZEKATZE
55 DATABIMMELBAHN
56 DATAQUARZUHR
57 DATAKOMPLIMENT
58 DATAKONDENSMILCH
59 DATAKOMBINATION
60 DATAREGENMANTEL
61 DATAKRANKENWAGEN
62 DATAKATASTROPHE
63 DATATOLPATSCH
64 DATATALISMANN
65 DATAHAGEBUTTE
66 DATAHASCHISCH
67 DATAHASCHEE
68 DATAINGENIEUR
69 DATAJOURNALIST
70 DATAKASSETTE
71 DATATEENAGER
72 DATATACHOMETER
73 DATAGRILLPARTY
74 DATASCHMALZLOCKE
75 DATAMITTELSCHEITEL
76 DATASTRICKNADEL
77 DATAEXPORTEUR
78 DATAUNTERGANG
79 DATAAMEISENHAUFEN
80 DATAPLUMBSKLO
81 DATAZAHNPASTA
82 DATAPICKNICKKORB
83 DATAZINNOBER
84 DATAZENTIMETER
85 DATATORNISTER
86 DATASEIFENSCHAUM
87 DATATHUNFISCH
88 DATABRILLENGESTELL
89 DATAZYANKALI
90 DATATOPFLAPPEN
91 DATABLAUBEERTORTE
92 DATASPRINTERHOSE
93 DATAHUNDEKNOCHEN
94 DATABRIEFTAUBE
95 DATATINTENPATRONE
96 DATAQUELLWASSER
97 DATAKASSENBON
98 DATAPETROLEUM
99 DATAPETERSILIE
100 DATAREGENSCHAUER
101 DATAPULLOVER
102 DATAPLANSCHBECKEN
103 DATAGIROKONTO
104 DATAEXPERIMENT
105 DATAVOLLMILCH
106 DATAPOLTERABEND
107 DATASTANDESAMT
108 DATASPRICHWORT
109 DATATUBERKULOSE
110 DATAREKLAMESENDUNG
111 DATAREAGENZGLAS
112 DATAORCHIDEE
113 DATAPAMPELMUSE
114 DATAZITRUSFRUCHT
115 DATAPANTOFFEL
116 DATAJUSTIZIRRTUM
117 DATACOWBOYFILM
118 DATACURRYSAUCE
119 DATAVOLLEYBALL
120 DATADSCHUNKE
121 DATADSCHUNGELTIER
122 DATAGASTRONOMIE
123 DATAEIDECHSE
124 DATABIMSSTEIN
125 DATABISAMRATTE
126 DATABASKETBALL
127 DATAPURZELBAUM
128 DATARANGIERBAHNHOF
129 DATAQUASSELEI
130 DATAQUACKSALBER
131 DATAQUADRATMETER
132 DATARAUCHVERBOT
133 DATAREPRODUKTION
134 DATASABOTEUR
135 DATARODELBAHN
136 DATASALMIAKGEIST
137 DATASAMARITER
138 DATASANATORIUM
139 DATASANDWICH
140 DATASCHLAFITTCHEN
141 DATASCHASCHLIK
142 DATASCHABERNACK
143 DATASAXOPHON
144 DATAHINRICHTUNG
145 DATASCHIMPANSE
146 DATASCHLAMPEREI
147 DATAVEGETARIER
148 DATAUTENSILIE
149 DATAAKKORDARBEIT
150 DATAALLHEILMITTEL
151 DATAANALPHABET
152 DATAANGORAKATZE
153 DATAAPPARTMENT
154 DATAAQUARIUM
155 DATAASTRONAUT
156 DATAAUSRUFEZEICHEN
157 DATABAGATELLE
158 DATABARIKADE
159 DATABAZILLUS
160 DATACHARAKTER
161 DATACHRYSANTHEME
162 DATACONTAINER
163 DATAFARBFERNSEHER
164 DATABRANNTWEIN
165 DATABUNSENBRENNER
166 DATABUMERANG
167 DATABUCHSTABE
168 DATABUDDHISMUS
169 DATABRITANNIEN
170 DATABEGLAUBIGUNG
171 DATAGARDEROBE
172 DATATRAUBENZUCKER
173 DATAEINSCHREIBEN
174 DATABRIEFDRUCKSACHE
175 DATAELEKTRIKER
176 DATASCHNELLBAHN
177 DATASCHLAMASSEL
178 DATASCHLARAFFENLAND
179 DATASCHNEEBALL
180 DATASCHOKOLADE
181 DATASCHMETTERLING
182 DATATRAUBENSAFT
183 DATAOZEANDAMPFER
184 DATAORGANISATOR
185 DATANACHWUCHS
186 DATAMURMELTIER
187 DATANACHMITTAG
188 DATAMAGNETBAND
189 DATAMANDARINE
190 DATAMANSARDE
191 DATAMANUSKRIPT
192 DATAMARATHONLAUF
193 DATAMARIONETTE
194 DATAMARMELADE
195 DATAMAKKARONI
196 DATAMEERRETTICH
197 DATAMETTWURST
198 DATARECHENSCHAFT
199 DATARANDALIERER
200 DATAREDAKTEUR
201 DATASADISMUS
202 DATAVERABREDUNG
203 DATARESIDENZ
204 DATARESTAURATOR
205 DATAREVANCHE
206 DATAREVOLUTION
207 DATARHYTMUS
208 DATARIZINUSOEL
209 DATAROLLBRATEN
210 DATAZUCKERROHR
211 DATASABOTAGE
212 DATAWILDSCHWEIN
213 DATASCHABLONE
214 DATASCHAUSPIELER
215 DATASCHIEDSRICHTER
216 DATASCHLAMPEREI
217 DATASCHORNSTEIN
218 DATALEBENSMITTEL
219 DATASCHREBERGARTEN
220 DATASCHULBUS
221 DATASCHWIEGERELTERN
222 DATASELTERSWASSER
223 DATAMILCHSHAKE
224 DATASENSATION
225 DATASKANDINAVIEN
226 DATASMARAGD
227 DATASPHINX
228 DATASTEWARD
229 DATAVERABREDUNG
230 DATAALUMINIUMFOLIE
231 DATASYMPHONIE
232 DATAKREPPAPIER
233 DATASCHICKSALSSCHLAG
234 DATAUMLAUFBAHN
235 DATAUREINWOHNER
236 DATAWAHRZEICHEN
237 DATAWEGERICH
238 DATAWEIZENBROT
239 DATAWEINBRAND
240 DATABRANNTWEIN
241 DATASCHUCREME
242 DATASEEFAHRERVOLK
243 DATAZAPFENSTREICH
244 DATAZITADELLE
245 DATAZEPPELIN
246 DATAZERVELATWURST
```

Listing 2. Hauptprogramm zu »Galgenraten«

```
247 DATASTUNDENPLAN
248 DATABILDERBUCH
249 DATAESSENSZEIT
250 DATATOMATENSALAT
251 DATAHAUSKATZE
252 DATAKANARIENVOGEL
253 DATAFLASCHENKORKEN
254 DATAHOLZGESTELL
255 DATATASCHENDIEB
256 DATA704,32,704,16,685,16,643,32,571,
32,596,32,643,32,1020,14,643,16,571,24
257 DATA596,8,685,32,643,32,704,32,739,3
2,1020,14,704,16,685,16,704,16,739,16
258 DATA704,16,685,32,1020,14,685,16,704
,16,685,16,643,32,571,32,596,32,643,32
259 DATA1020,14,643,16,571,24,596,8,685,
32,643,32,704,32,739,32,1020,14,739,16
260 DATA704,16,685,16,643,32,596,32,685,
16,704,32,739,16,704,48,1020,64,-1
261 DATA1020,10,881,5,881,15,881,5,917,2
0,917,20,929,20,929,20,953,30,939,10
262 DATA917,10,953,5,939,15,917,5,897,20
,944,40,929,15,911,5,917,40,1020,64,-1,
263 FORA=1TO7
264 KEYA,""
265 NEXT
266 KEY8,"←"
267 POKE51,0:POKE52,60:POKE55,0:POKE56,6
0:POKE216,0:POKE217,0:POKE1344,1
268 POKE65286,11:POKE65298,8:POKE65299,6
0
269 COLOR4,1:COLOR0,1
270 PRINT"{2HOME,CLR,CTRL-H,CTRL-N}"
271 COLOR1,16,5
272 CHAR1,14,0,"←[#←]%'[)]%"
273 CHAR1,14,1,CHR$(34)+"£$"+CHR$(34)+"↑
&(£*↑&"
274 COLOR1,3,5
275 CHAR1,0,6,"BXTKU I*HSYU= SXU:"
276 CHAR1,3,8,"ZT* C=±↑TKU*OT TOHZ TU  G
T U**OKU=TU="
277 CHAR1,3,9,"W=*KU !"
278 CHAR1,3,11,"ZT* EX=VORU TU  GT U**OK
U=TU= W=*KU "
279 CHAR1,3,12,"IT*SH UX=U O=TU*U PU* ==
!"
280 COLOR1,2,4
281 CHAR1,0,8,"{RVSON}F1"
282 CHAR1,0,11,"{RVSON}F2{RVOFF}"
283 COLOR1,2,1
284 CHAR1,22,24,"MOTU RG STU= JT=V"
285 POKE65286,27
286 IFPEEK(198)=4THEN359
287 IFPEEK(198)=5THEN289
288 GOTO286
289 COLOR1,2,4
290 CHAR1,0,11,"{RVSON,FLASH ON}F2{RVOFF
}"
291 FORA=1023TO0STEP-8
292 VOLA/128
293 SOUND1,A,1
294 NEXT
295 POKE65286,11
296 PRINT"{2HOME,CLR,CTRL-H,CTRL-N}"
297 COLOR1,16,5
298 CHAR1,14,0,"←[#←]%'[)]%"
299 CHAR1,14,1,CHR$(34)+"£$"+CHR$(34)+"↑
&(£*↑&"
300 COLOR1,3,5
301 CHAR1,0,6,"BXTKU ZO  U= SXU  XSH UX=
 W=*K UX=VURU=!"
302 CHAR1,0,8,"WU== SXU UX=U= FUHZU* HOR
U=, IO== ±H  U="
303 CHAR1,0,9,"SXU TXU TO KU{6SPACE}I*HS
YU=!"
304 CHAR1,0,11,"WU== IO  W=*K UXVUVURU=
X K, IO== ±H  U="
305 CHAR1,0,12,"SXU TXU TO KU{8SPACE}I*H
SYU=!"
306 COLOR1,2,4
307 CHAR1,14,9,"{RVSON}HELP"
308 CHAR1,14,12,"{RVSON}RETURN{RVOFF}"
309 COLOR1,2,1
310 CHAR1,22,24,"MOTU RG STU= JT=V"
311 POKE65286,27
312 COLOR1,8,6
313 CHAR1,12,16,"{FLASH ON}@{FLASH OFF}@
@@@@@@@@@@@@@"
314 CLR
315 VOL6
316 POKE239,0
317 GETA$
318 IFA$="←"ANDLEN(B$)>0THEN329
319 IFA$=CHR$(13)ANDA>4THEN335
320 IFASC(A$)>90ORASC(A$)<65THEN317
321 SOUND1,836,4
322 A=A+1
323 IFA=16THEN312
324 CHAR1,11+A,16,A$
325 B$=B$+A$
326 IFA=15THEN316
327 CHAR1,12+A,16,"{FLASH ON}@{FLASH OFF
}"
328 GOTO316
329 B$=LEFT$(B$,LEN(B$)-1)
330 A=A-1
331 CHAR1,12+A,16,"{FLASH ON}@{FLASH OFF
}"
332 IFA<14THENPRINT"@"
333 SOUND1,836,4
334 GOTO316
335 POKE65286,11
336 PRINT"{2HOME,CLR,CTRL-H,CTRL-N}"
337 COLOR1,16,5
338 CHAR1,14,0,"←[#←]%'[)]%"
339 CHAR1,14,1,CHR$(34)+"£$"+CHR$(34)+"↑
&(£*↑&"
340 COLOR1,3,5
341 CHAR1,0,6,"BXTKU I*HSYU= SXU:"
342 CHAR1,3,8,"WU== IO  W=*K *XSHKXV"
343 CHAR1,3,9,"VU SH*XURU= X K!"
344 CHAR1,3,11,"WU== IO  W=*K =XSHK *XSH
KXV"
345 CHAR1,3,12,"VU SH*XURU= X K!"
346 COLOR1,2,4
347 CHAR1,0,8,"{RVSON}F1"
348 CHAR1,0,11,"{RVSON}F2{RVOFF}"
349 COLOR1,8,6
350 CHAR1,12,16,"@@@@@@@@@@@@@@@"
351 CHAR1,12,16,B$
352 COLOR1,2,1
353 CHAR1,22,24,"MOTU RG STU= JT=V"
354 POKE65286,27
355 POKE239,0
356 IFPEEK(198)=5THEN291
357 IFPEEK(198)<>4THEN356
358 GOTO363
359 COLOR1,2,4
360 CHAR1,0,8,"{RVSON,FLASH ON}F1{RVOFF}
"
361 RESTOREINT(RND(1)*314159265)AND255
362 READB$
```

```
363 FORA=1023TO0STEP-8
364 VOLA/128
365 SOUND1,A,1
366 NEXT
367 POKE65286,11
368 PRINT"{2HOME,CLR,CTRL-H,CTRL-N}"
369 COLOR1,16,5
370 CHAR1,14,0,"←[#←]%'[)]%"
371 CHAR1,14,1,CHR$(34)+"£$"+CHR$(34)+"↑
&(£*↑&"
372 COLOR1,3,5
373 CHAR1,1,4,"BXKKU VURU= SXU IHKU BTSH
 KQRU= UX=!"
374 COLOR1,8,6
375 FORB=0TOLEN(B$)-1
376 CHAR1,(28-LEN(B$))/2+B,10,"@"
377 NEXT
378 COLOR1,2,1
379 CHAR1,22,24,"MQTU KG STU= JT=V"
380 POKE65286,27
381 C$=B$
382 A=1
383 VOL6
384 DO
385 POKE239,0
386 GETKEYA$
387 IFASC(A$)>90ORASC(A$)<65ORINSTR(D$,A
$)>0THENLOOP
388 SOUND1,836,5
389 D$=D$+A$
390 IFINSTR(B$,A$)=0THENONAGOTO400,409,4
17,425,432,439,447,455
391 COLOR1,8,6
392 CHAR1,(28-LEN(B$))/2-1+INSTR(C$,A$),
10,A$
393 MID$(C$,INSTR(C$,A$),1)=" "
394 C=C+1
395 IFINSTR(C$,A$)>0THEN392
396 IFC<LEN(B$)THENLOOP
397 RESTORE261
398 POKE217,(PEEK(217)+1)AND255
399 GOTO465
400 COLOR1,8,6
401 CHAR1,0,16,"FQZ SHU BTSH KQRU=:"
402 COLOR1,3,3
403 CHAR1,20,16,A$
404 COLOR1,6,1
405 CHAR1,27,19,"+---------/"
406 CHAR1,26,20,"+-----------/"
407 A=2
408 LOOP
409 COLOR1,3,3
410 CHAR1,21,16,A$
411 COLOR1,10,1
412 FORA=7TO18
413 CHAR1,35,A,"-"
414 NEXT
415 A=3
416 LOOP
417 COLOR1,3,3
418 CHAR1,22,16,A$
419 COLOR1,10,1
420 CHAR1,29,7,"------{DOWN,LEFT};"
421 COLOR1,8,3
422 CHAR1,29,8,"<="
423 A=4
424 LOOP
425 COLOR1,3,3
426 CHAR1,23,16,A$
427 COLOR1,9,4
428 CHAR1,29,9,">*"
429 CHAR1,29,10,"AB"
430 A=5
431 LOOP
432 COLOR1,3,3
433 CHAR1,24,16,A$
434 CHAR1,29,11,"CD"
435 CHAR1,29,12,"-E"
436 CHAR1,29,13,"-E"
437 A=6
438 LOOP
439 COLOR1,3,3
440 CHAR1,25,16,A$
441 CHAR1,28,11,"F"
442 CHAR1,28,12,"H"
443 CHAR1,28,13,"J"
444 CHAR1,28,14,"K"
445 A=7
446 LOOP
447 COLOR1,3,3
448 CHAR1,26,16,A$
449 CHAR1,31,11,"G"
450 CHAR1,31,12,"I"
451 CHAR1,31,13,"J"
452 CHAR1,31,14,"K"
453 A=8
454 LOOP
455 COLOR1,7,1
456 CHAR1,29,14,"LM"
457 CHAR1,29,15,"NO"
458 CHAR1,29,16,"NO"
459 COLOR1,10,0
460 CHAR1,29,17,"PP"
461 COLOR1,8,6
462 CHAR1,(28-LEN(B$))/2,10,"{FLASH ON}"
+B$
463 VOL8
464 RESTORE256
465 DO
466 READA,B
467 IFA=-1THENEXIT
468 SOUND1,A,B
469 SOUND1,1020,2
470 LOOP
471 POKE65286,11
472 POKE216,(PEEK(216)+1)AND255
473 PRINT"{2HOME,CLR,CTRL-H,CTRL-N}"
474 COLOR1,16,5
475 CHAR1,14,0,"←[#←]%'[)]%"
476 CHAR1,14,1,CHR$(34)+"£$"+CHR$(34)+"↑
&(£*↑&"
477 COLOR1,3,5
478 CHAR1,2,6,"BX HU* TQ*U="+STR$(PEEK(2
16))+" W**KU* GI U**QKU=."
479 CHAR1,2,8,"E  TI*IU="+STR$(PEEK(217)
)+" W**KU* U**QKU=."
480 CHAR1,6,13,"M*SHKU= SXU TUXKU*  ↑XUZ
U= ?"
481 CHAR1,12,15,"JQ -TU* NUX= ?"
482 POKE65286,27
483 POKE239,0
484 GETKEYA$
485 IFA$<>"N"THENRUN268
486 SYS65529
```

Listing 2. Hauptprogramm »Galgenraten« (Schluß). Bitte beachten Sie die Eingabehinweise auf Seite 76/77.

Kampf um Rom

Imperium Romanum ist ein Strategiespiel für zwei Personen, das sich besonders durch seine sehr gute Bildschirmgestaltung auszeichnet. Es zeigt, daß auch auf dem VC 20 das letzte Wort über Grafik noch nicht gesprochen ist.

Das Spiel besteht aus einer Europakarte (siehe Bild), 20 römischen Legionen, 15 germanischen Horden und einem Fadenkreuz (zur Bewegung dieser Legionen).

Jede dieser Einheiten kann pro Runde (entspricht einem Jahr) zwei »Schritte« in eine beliebige Richtung marschieren, sofern diese Felder frei sind und nicht im Meer liegen.

Sinn des Spiels

Anfangs wird gelost, welcher Spieler die Germanen und wer die Römer übernimmt. Die Germanen müssen, um das Spiel siegreich zu beenden, mindestens vier der fünf Römerstädte erobern (das heißt sich auf deren Feld stellen). Die Aufgabe der Römer ist es, dies 25 Jahre lang zu verhindern.

Die Römer sollten sich jedoch keinesfalls von der geringen Anzahl der Germanenhorden zu unüberlegten Taten hinreißen lassen. Die Kampfkraft der Germanen ist zu Beginn des Spiels wesentlich höher als die der Römer. Deshalb sollten die Römer zuerst die Taktik des geordneten Rückzugs anwenden, aber auch darauf achten, daß dieser nicht in eine kopflose Flucht ausartet.

Im Jahre 405 nach Christus wird sich dann das Blatt wenden. In diesem Jahr erhalten alle noch vorhandenen Einheiten Nachschub, aber dieses Mal bekommen die Römer ein größeres Stück von dem Kuchen ab.

Das Spiel ist ebenfalls beendet, wenn einer der beiden Spieler keine Einheiten mehr zur Verfügung hat.



Bild. Eine aufwendige Grafik zeichnet dieses Strategiespiel aus

Kämpfe

Eine Kampfsituation entsteht, wenn zwei oder mehr feindliche Einheiten auf angrenzenden Feldern stehen. (Bei einem solchen Kampf wird jeder teilnehmenden Einheit ein wenig von ihrer Kampfstärke abgezogen. Sollte die Kampfstärke auf Null abgesunken sein, so löst sich die betreffende Einheit auf.) Da diese kämpfenden Einheiten dann jedes halbe Jahr erneut auf sich einprügeln, wenn sie nicht versetzt werden, sollte es Sie nicht wundern, wenn nach einigen Jahren mehrere dieser Legionen das Zeitliche segnen und von der Landkarte verschwinden.

Strategie

Wenn man sich auf solche Kämpfe einläßt, so sollte man eine feindliche Einheit mit mehreren eigenen Legionen umstellen, damit man sich sicher sein kann, dieses Gefecht als Sieger zu überstehen. Es ist zu empfehlen, die Legionen mit häufigem Feindkontakt auszuwechseln, damit die eigenen Einheiten nicht aufgerieben, sondern nur geschwächt werden (im Jahre 405 werden sie dann wieder gestärkt).

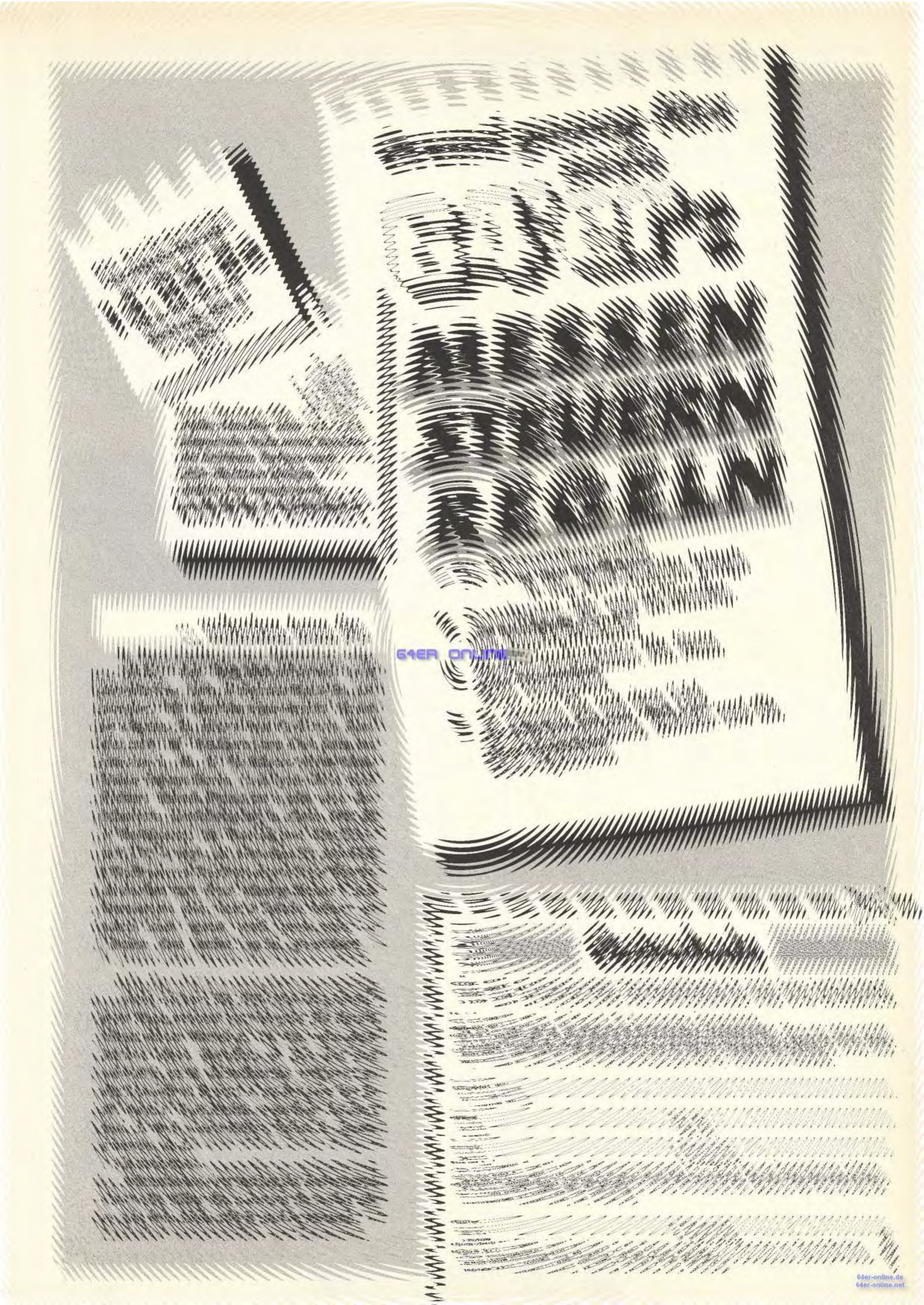
Die Römer sollten sich auf die Verteidigung von zwei bis drei Städten beschränken, um nicht von den germanischen Horden, die sich darum reißen, an Odins Tafel teilzuhaben, überrannt zu werden.

Steuerung

Der Spieler, der Rom vertritt, beginnt das Spiel. Mit dem Joystick kann er das Fadenkreuz bewegen. Das Fadenkreuz läßt sich weder aus dem Bildschirm heraus- noch durch das Meer steuern und hat somit die gleichen Eigenschaften wie die Legionen, die man bewegen möchte. Aus diesem Grund sollte man zuerst mit dem Fadenkreuz die Umgebung ab-

Variable	Bedeutung
O1,O2	= Bildschirmfarben
JA	= Jahreszahl
F	= Vordergrund
M,N	= Absperrung des Bildrandes für bewegliche Einheiten
A(I)	= Römer-Legionen (Türme)
B(I)	= Germanen-Horden (Hütten)
K1(I)	= Kampfkraft der Römer
K2(I)	= Kampfkraft der Germanen
BW(I)	= Anzahl der Bewegungen einer Einheit pro Jahr
K(I)	= Position der Städte
FA	= Fadenkreuz
P1	= Kartenuntergrund unter dem Fadenkreuz
P2,P3,P4,P5	= Umgebung des Fadenkreuzes
Z	= Abwechseln der Spieler
U1,U2,U3,U4	= Bewegungsrichtungen des Fadenkreuzes
SC	= Feuertaste
W2,W3,W4,W5	= Umgebung der Legionen
X	= Position der eben bewegten Einheit
ZT	= »Jahre« seit Spielbeginn
TR,TL	= Verluste der Einheiten pro Kampf

Tabelle 1. Variablenliste



tasten, um festzustellen, wohin sich die Legionen bewegen können und wo nicht.

Hat man sich nun für eine bestimmte Richtung entschieden, so setzt man das Fadenkreuz auf die fortzubewegende (eigene) Einheit und drückt die Feuertaste. Mit einem kurzen

	Hauptprogramm
15 - 98	Initialisierung der Variablen
110 - 190	Erstellen der Europakarte
200 - 220	Abdruck der Römer-, Germanen- und Städtesymbole auf dem Bildschirm
300 - 310	Eventuelle Abgabe an das Unterprogramm
320	Aufruf der Joystickabfrage
325 - 375	Darstellung und Bewegung des Fadenkreuzes
380 - 450	Tastaturabfrage
	Unterprogramme
480 - 610	Bewegung der Legionen
615 - 620	Kontakt einer Germanen-Horde mit einer Stadt? Sollten vier Städte zerstört sein, dann nach 2000 verzweigen.
630 - 700	Bei Druck auf Feuertaste erfolgt Rücksprung ins Hauptprogramm
800 - 803	Ablauf der Jahre
805	Nach 25 »Jahren« endet das Programm
810 - 1010	Gefecht. Abzüge von der Kampfkraft
1015 - 1020	Ausgabe der Jahreszahl
2000 - 2150	Sieg der Germanen
2500 - 2640	Sieg der Römer
3000 - 3040	Joystickabfrage
5000 - 5030	Änderung der Bildschirmfarben über Tasten

Tabelle 2. Programmbeschreibung

Piepston wandelt sich das Kreuz in die darunter befindliche Einheit um. Nun kann diese Legion mit Hilfe des Joysticks bis zu zwei Schritte weit in die gewünschte Richtung marschieren. (Wenn diese Richtung nicht bereits von einer anderen Einheit blockiert wird. Man kann weder eine eigene noch eine fremde Legion »überspringen«.)

Wenn Sie mit Ihrem Steuermanöver fertig sind, müssen Sie erneut die Feuertaste betätigen. Das Fadenkreuz wird dadurch wieder frei.

Auf diese Art und Weise können Sie alle Ihre Einheiten (sofern diese nicht blockiert sind) versetzen. Wenn Sie alle Einheiten, die Sie bewegen wollten, versetzt haben, müssen Sie eine Taste drücken. Dadurch beginnen die feindlichen Legionen ihr Gefecht, und man »hört« ein halbes Jahr vergehen: Nachdem die Melodie verklungen ist, wird der Joystick weitergereicht, und der zweite Spieler ist an der Reihe. Interessierte Leser finden in den Tabellen 1 und 2 eine Variablenbelegung und eine Dokumentation zum Programm.

(Ralf Trabhardt/tr)

Wichtig !

Beachten Sie bitte, daß Ihr VC 20 mindestens eine 8-KByte-Speichererweiterung besitzen muß. Tippen Sie zuerst Listing 1 ein, speichern Sie es, und dann Listing 2 (ebenfalls speichern!) Wenn Sie spielen wollen, müssen Sie *vor* dem Laden des Vorprogramms folgendes eintippen:

`POKE 44,28 : POKE 7168,0 : NEW`


```
5 PRINT"{CLR,4DOWN,RIGHT}BITTE 7 SEC. WARTEN!"  <040>
6 FOR I=5120 TO 6423:READ K:S=S+K:NEXT:IF S<>14
  2823 THEN PRINT"FEHLER IN DATAS!":STOP        <073>
7 REM FORI=6400TO6407:POKEI,255:NEXT            <253>
8 PRINT"{DOWN}LADEN SIE NUN DAS{5SPACE,DOWN}HAU
  PTPROGRAMM!":END                              <187>
10 DATA 255, 255, 255, 255, 255, 255, 255, 255  <092>
11 DATA  0, 7, 31, 31, 63, 63, 127, 255         <207>
12 DATA 15, 63, 31, 255, 255, 255, 255, 255     <104>
13 DATA 48, 60, 126, 63, 191, 191, 191, 255     <028>
14 DATA 0, 176, 240, 240, 252, 252, 253, 255    <202>
15 DATA 8, 30, 7, 19, 15, 15, 15, 1             <135>
16 DATA 255, 255, 127, 63, 31, 31, 31, 15       <094>
17 DATA 7, 31, 15, 7, 7, 7, 7, 3                <151>
18 DATA 3, 1, 1, 1, 3, 3, 1, 5                  <124>
19 DATA 5, 13, 14, 15, 31, 31, 31, 31           <135>
20 DATA 31, 63, 63, 127, 127, 255, 255, 255     <095>
21 DATA 0, 0, 0, 224, 251, 255, 255, 255        <126>
22 DATA 224, 240, 254, 255, 255, 255, 255, 255  <237>
23 DATA 0, 0, 0, 152, 252, 252, 254, 255        <099>
24 DATA 127, 15, 1, 0, 0, 0, 0, 0               <099>
25 DATA 255, 255, 255, 255, 255, 254, 254, 248  <078>
26 DATA 254, 240, 224, 192, 192, 192, 0, 0      <191>
27 DATA 254, 254, 254, 252, 252, 254, 254, 255  <163>
28 DATA 254, 248, 224, 192, 192, 128, 128, 0    <099>
29 DATA 255, 255, 255, 255, 255, 255, 255, 254  <109>
30 DATA 255, 255, 255, 255, 254, 252, 120, 0    <210>
31 DATA 255, 252, 248, 248, 240, 240, 224, 240  <171>
32 DATA 248, 248, 240, 224, 0, 0, 0, 0          <203>
33 DATA 191, 63, 31, 7, 0, 0, 0, 0              <199>
34 DATA 255, 255, 254, 252, 248, 112, 0, 0      <016>
35 DATA 255, 255, 255, 255, 255, 255, 253, 252  <047>
36 DATA 255, 255, 255, 255, 255, 255, 255, 127  <246>
37 DATA 31, 7, 1, 1, 1, 3, 3, 1                 <190>
38 DATA 255, 255, 255, 255, 255, 127, 63, 31    <120>
39 DATA 31, 15, 7, 7, 3, 1, 1, 0                <217>
40 DATA 240, 248, 248, 252, 252, 252, 254, 254  <146>
41 DATA 128, 128, 128, 192, 192, 224, 224, 240  <068>
42 DATA 0, 0, 0, 0, 0, 0, 0, 0                  <216>
43 DATA 255, 255, 127, 39, 7, 7, 3, 3           <071>
44 DATA 255, 255, 171, 131, 131, 147, 147, 255  <141>
45 DATA 128, 0, 0, 128, 224, 240, 248, 255      <074>
46 DATA 255, 255, 255, 249, 248, 240, 240, 240  <018>
47 DATA 128, 192, 224, 224, 112, 48, 48, 32     <100>
48 DATA 224, 248, 248, 252, 252, 252, 124, 96   <222>
49 DATA 224, 224, 224, 192, 128, 0, 0, 0        <164>
50 DATA 1, 1, 1, 1, 91, 243, 240, 224           <238>
51 DATA 224, 224, 224, 224, 224, 224, 192, 0    <190>
52 DATA 127, 63, 31, 3, 1, 1, 0, 0              <044>
53 DATA 0, 0, 0, 160, 224, 252, 255, 255        <053>
54 DATA 0, 0, 1, 1, 1, 3, 3, 1                  <081>
55 DATA 192, 128, 128, 192, 240, 248, 248, 248  <052>
56 DATA 128, 128, 196, 220, 252, 248, 248, 240  <107>
57 DATA 191, 255, 255, 255, 255, 255, 255, 255  <067>
58 DATA 0, 0, 0, 0, 0, 0, 12, 159              <102>
59 DATA 0, 0, 0, 0, 0, 0, 12, 252              <251>
60 DATA 0, 0, 0, 0, 0, 0, 32, 61               <202>
62 DATA 0, 0, 0, 0, 248, 252, 254, 254          <103>
63 DATA 0, 0, 0, 0, 62, 255, 255, 255           <165>
64 DATA 0, 0, 0, 240, 255, 255, 255, 255        <147>
65 DATA 0, 0, 0, 0, 0, 71, 239, 255             <117>
66 DATA 0, 0, 0, 0, 0, 7, 31, 63                <191>
67 DATA 0, 0, 0, 0, 208, 240, 252, 255          <233>
68 DATA 192, 128, 131, 131, 129, 129, 128, 128  <053>
69 DATA 255, 255, 255, 255, 255, 255, 255, 243  <146>
70 DATA 255, 255, 255, 191, 159, 143, 175, 167  <162>
71 DATA 7, 23, 11, 3, 23, 3, 3, 1               <137>
72 DATA 255, 127, 47, 7, 3, 9, 40, 3            <012>
73 DATA 1, 0, 0, 0, 0, 0, 0, 0                  <255>
74 DATA 255, 63, 31, 15, 7, 3, 1, 0             <147>
75 DATA 127, 63, 63, 127, 255, 255, 127, 255    <022>
76 DATA 255, 127, 127, 255, 127, 63, 31, 15     <169>
77 DATA 7, 7, 7, 3, 3, 1, 1, 0                  <084>
78 DATA 127, 255, 255, 127, 0, 10, 63, 127      <071>
79 DATA 0, 2, 0, 0, 4, 2, 0, 0                  <157>
80 DATA 63, 63, 31, 31, 31, 31, 29, 9           <036>
81 DATA 1, 1, 0, 0, 0, 0, 0, 0                  <039>
82 DATA 176, 144, 0, 32, 0, 0, 0, 0             <153>
83 DATA 241, 240, 248, 208, 192, 192, 224, 224  <092>
84 DATA 204,198,227,253,126,62,191,255          <035>
85 DATA 0,0,128,128,192,64,32,0                 <046>
86 DATA 128,192,192,224,224,224,192,192         <033>
87 DATA 254,254,255,191,63,188,156,150          <132>
88 DATA 0, 0, 0, 128, 128, 0, 0, 0              <254>
89 DATA 255, 255, 255, 255, 255, 255, 249, 240  <017>
90 DATA 0, 0, 0, 128, 224, 240, 248, 255        <184>
91 DATA 28, 26, 51, 39, 15, 31, 31, 63          <105>
92 DATA 3,127,255,255,255,255,255,255           <006>
93 DATA 0,198,255,255,127,255,255,255           <052>
94 DATA 224,252,252,249,224,128,1,3             <151>
```

Listing 1. Das Vorprogramm zu »Imperium Romanum« enthält die Informationen zum geänderten Zeichensatz

```
95 DATA 128,128,192,224,224,192,192,224          <047>
96 DATA 240,128,0,0,0,128,128,128                <103>
97 DATA 240,240,240,240,240,240,240,240          <199>
98 DATA 255,255,254,254,244,240,240,240          <156>
99 DATA 224,192,128,128,0,0,0,0                  <059>
100 DATA 255,255,255,254,252,240,224,224         <244>
101 DATA 63,103,15,103,39,7,7,15                 <018>
102 DATA 3,3,223,207,79,65,0,0                   <190>
103 DATA 255,127,63,255,7,7,31,28                <023>
104 DATA 255,255,255,255,255,255,191,63          <008>
105 DATA 0,28,28,28,16,0,0,0                     <251>
106 DATA 0,0,0,0,0,0,64,0                        <219>
107 DATA 0,0,0,0,0,0,0,0                         <025>
108 DATA 0,4,0,0,0,2,2,0                         <219>
109 DATA 0,0,1,3,7,1,129,192                     <163>
110 DATA 0,0,135,199,192,128,128,0               <017>
111 DATA 0,0,0,1,3,3,3,3                         <030>
112 DATA 3,3,3,3,3,1,0,1                         <087>
113 DATA 192,128,128,128,128,0,0,0               <017>
114 DATA 0,128,128,128,128,128,128,0             <190>
115 DATA 49,55,63,63,31,63,63,63                 <083>
116 DATA 31,31,31,63,63,63,59,49                 <043>
117 DATA 0,128,128,128,128,0,0,0                 <201>
118 DATA 0,0,0,0,0,0,32,48                       <052>
119 DATA 240,224,192,0,0,0,0,0                   <250>
120 DATA 248,0,0,0,0,0,0,0                       <233>
121 DATA 239,199,239,199,187,187,199,255         <074>
122 DATA 63,63,63,31,15,15,1,0                   <221>
123 DATA 62,6,0,0,0,0,0,0                        <032>
124 DATA 1,65,65,192,192,192,192,192             <084>
125 DATA 231,231,231,0,0,231,231,231             <100>
126 DATA 255,255,231,195,129,153,255,255         <119>
127 DATA 255,255,251,227,195,147,147,255         <052>
128 DATA 11,205,255,255,255,255,255,255          <124>
129 DATA 0,1,3,3,1,1,7,7                         <143>
130 DATA 8,32,79,7,71,67,195,231                 <006>
131 DATA 24,63,255,255,255,255,255,255           <004>
132 DATA 127,255,255,255,255,255,255,255         <222>
133 DATA 124,124,62,62,127,127,255,63            <048>
134 DATA 124,252,252,60,56,61,61,60              <118>
135 DATA 15,31,63,63,127,127,127,124             <146>
136 DATA 192,192,128,192,192,128,0,0             <043>
137 DATA 4,30,30,156,220,220,134,0               <246>
138 DATA 0,48,32,0,0,152,188,255                 <215>
139 DATA 180,240,249,249,249,251,255,255         <059>
140 DATA 3,7,191,255,255,255,255,255             <077>
141 DATA 0,0,0,0,0,0,0,1                         <061>
142 DATA 0,0,0,0,24,56,248,254                   <149>
143 DATA 7,7,39,39,63,63,63,127                  <131>
144 DATA 3,3,3,3,1,1,3,3                         <236>
145 DATA 0,0,0,1,1,1,3,3                         <240>
146 DATA 63,60,60,60,120,60,12,0                 <223>
147 DATA 255,255,127,127,255,255,255,127         <237>
148 DATA 254,252,252,252,252,253,248,224         <066>
149 DATA 0,0,0,0,128,128,0,0                     <060>
150 DATA 224,0,0,0,0,0,0,0                       <102>
151 DATA 255,255,239,207,159,147,255,255         <068>
160 DATA 48, 48, 24, 24, 12, 12, 6, 6            <026>
161 DATA 0, 137, 205, 116, 84, 68, 36, 34        <157>
162 DATA 0, 243, 10, 138, 146, 227, 129, 65      <044>
163 DATA 0, 231, 4, 4, 4, 199, 5, 5              <218>
164 DATA 0, 130, 66, 68, 68, 132, 8, 8           <081>
165 DATA 0, 17, 33, 33, 34, 66, 68, 69           <025>
166 DATA 0, 66, 102, 108, 180, 168, 136, 16      <191>
167 DATA 6, 3, 0, 0, 0, 0, 1, 2                  <105>
168 DATA 34, 18, 0, 0, 0, 0, 240, 35             <125>
169 DATA 65, 65, 0, 0, 0, 0, 0, 137              <237>
170 DATA 4, 196, 0, 0, 0, 0, 0, 24               <098>
171 DATA 136, 136, 0, 0, 0, 0, 0, 144            <101>
172 DATA 73, 49, 0, 0, 0, 0, 0, 12               <100>
173 DATA 16, 16, 0, 0, 0, 0, 0, 128              <042>
174 DATA 4, 7, 10, 18, 34, 33, 65, 65            <249>
175 DATA 68, 136, 16, 17, 33, 35, 38, 28         <059>
176 DATA 203, 213, 149, 161, 33, 33, 65, 65      <141>
177 DATA 36, 36, 60, 36, 66, 66, 66, 66          <227>
178 DATA 202, 201, 169, 164, 148, 138, 134, 131  <071>
179 DATA 0,0,0,0,0,0,0,0                         <097>
180 DATA 42, 21, 20, 138, 74, 69, 36, 24         <220>
181 DATA 192, 224, 160, 16, 16, 8, 132, 130      <160>
```

© 64'er

Listing 1. Das Vorprogramm zu »Imperium Romanum« Bitte beachten Sie die Eingabehinweise auf Seite 76.


```
1 REM********VC-20+8K*******                    <136>
2 REM*  IMPERIUM ROMANUM    *                    <166>
3 REM*   GESCHRIEBEN VON    *                    <123>
4 REM*   RALF TRABHARDT     *                    <047>
5 REM* PHILIPPSBERGSTR.45   *                    <009>
6 REM*   6200    WIESBADEN  *                    <155>
7 REM*  TEL. 06121/523970   *                    <160>
8 REM**********************                      <149>
9 PRINT"{CLR,BLACK}"                             <083>
10 POKE 650,128:O1=24:GOSUB 5000:POKE 650,
   0                                             <248>
15 F=33792:U=37151                               <127>
20 JA=395                                        <192>
25 POKE 36878,15                                 <197>
30 J=4108:M=4558:N=4579                          <150>
35 DIM A(26),B(26),K1(26),K2(26),BW(26)          <039>
40 FOR L=1 TO 20:READ A(L):NEXT L                <123>
45 FOR I=1 TO 15:READ B(I):NEXT I                <142>
50 FOR I=1 TO 5:READ K(I):NEXT I                 <030>
98 FOR I=1 TO 20:K1(I)=70:K2(I)=100:NEXT I       <110>
100 REM*ERSTELLEN DER KARTE*                     <254>
110 POKE 36869,205:PRINT"{CLR,BLACK,HOME,1
    0SPACE}CX{RVSON}HIJF{RVOFF}@@@@@@{RVSO
    N}MNOPQRS{RVOFF,3SPACE}FV{RVSON}GK{RVO
    FF,SPACE,RVSON}E{RVOFF}@@@@@@";              <033>
120 PRINT"{RVSON}TUVWXYZ{RVOFF,3SPACE}QB{S
    PACE,RVSON}BCD{RVOFF}@@@@@@";                <154>
125 PRINT"{RVSON}[£]↑←!{RVOFF,2SPACE}YUD@{
    RVSON}@A{RVOFF}@@@@@@@@{7SPACE}T@@@@@@
    @@@@@@@@";                                   <234>
130 PRINT"{4SPACE}G AJ@@@@@@@@@@@@@@@";          <064>
131 PRINT"{2SPACE}EDCB@@@@@@@@@@@@@@@@@";        <093>
132 PRINT"{3SPACE}F@@@@@@@@@@@@@@@@@@{3SPA
    CE}G@@@@@@@@@@@@@@@@@@{3SPACE}H@@@@@@@
    @@@@@@@@@@@";                                <013>
135 PRINT"M{2SPACE}I@@@@@@@@@@@@@@@@@Y@LKJ
    @@@@@@@:@@@@@@@@@X";                         <093>
140 PRINT"@@@@@@@@YZ@9;@@@@@@@@H @@@@@UWX7[
    @←<@@@@@@@V ";                               <130>
150 PRINT"@@@STV{2SPACE}@E£↑>=@@@@@@U @@@R
    {4SPACE}G+]@Q ?@@@@@T ";                     <163>
160 PRINT"@@Q TT{2SPACE}£M !@# *@@N@5R@OP{
    5SPACE}£N{2SPACE}P$%A@LMPQ@";                <228>
170 PRINT"ND{10SPACE}*& B@K Z@@876543120 ,
    +(' DCIJ+@@";                                <150>
180 PRINT"@@@@@@@@/.*){3SPACE}EHK*-@@@@@@
    @@@@-{6SPACE}FG{2SPACE}↑↑";                  <141>
181 FOR I=4580 TO 4601:POKE I,0:POKE I+F,O
    2:NEXT I                                     <240>
182 POKE 4168,162                                <023>
200 FOR I=1 TO 20:IF K1(I)>0 THEN POKE A(I
    ),34                                         <161>
201 IF K2(I)>0 AND B(I)>0 THEN POKE B(I),1
    15                                           <110>
205 NEXT I                                       <033>
210 FOR I=1 TO 5:IF K(I)>6 THEN POKE K(I),
    110:NEXT I                                   <246>
220 FA=4326:P1=PEEK(FA)                          <076>
300 REM                                          <108>
310 IF G1=1 OR G1=2 THEN 500                     <245>
320 GOSUB 3000                                   <038>
322 REM*BEWEGUNG DES FADENKREUZES*               <040>
325 P2=PEEK(FA-22):IF P2<>32 THEN SR=1           <189>
327 IF FA<4118 THEN SR=0                         <056>
330 IF U2=1 AND SR=1 THEN POKE FA,P1:FA=FA
    -22:P1=P2:GOTO 400                           <227>
335 P3=PEEK(FA+22):IF P3<>32 THEN SR=2           <208>
336 IF FA>4557 THEN SR=0                         <005>
340 IF U4=1 AND SR=2 THEN POKE FA,P1:FA=FA
    +22:P1=P3:GOTO 400                           <021>
```

Listing 2. »Imperium Romanum«, Hauptprogramm

```
345 P4=PEEK(FA-1):IF P4<>32 THEN SR=3            <075>
346 IF FA=M OR FA=M-22 OR FA=M-44 OR FA=M-
    66 OR FA=M-88 OR FA=M-198 OR FA=M-220
    OR FA=M-242 THEN SR=0                        <194>
350 IF U1=1 AND SR=3 THEN POKE FA,P1:FA=FA
    -1:P1=P4:GOTO 400                            <027>
355 P5=PEEK(FA+1):IF P5<>32 THEN SR=4            <220>
356 IF FA=N OR FA=N-22 OR FA=N-44 OR FA=N-
    66 OR FA=N-88 OR FA=N-110 OR FA=N-220
    OR FA=N-242 THEN SR=0                        <094>
357 IF FA=N-264 OR FA=N-418 OR FA=N-440 OR
     FA=N-462 THEN SR=0                          <174>
360 IF U3=1 AND SR=4 THEN POKE FA,P1:FA=FA
    +1:P1=P5:GOTO 400                            <076>
365 FOR I=1 TO 20:IF A(I)=FA AND Z=0 THEN
    F1=1:GOTO 370                                <031>
366 IF Z=1 AND B(I)=FA THEN F1=2:GOTO 370        <106>
367 NEXT I                                       <197>
370 IF SC=1 AND F1=1 THEN G1=1:F1=0:POKE 3
    6876,250:POKE 36876,0:GOTO 400               <004>
375 IF SC=1 AND F1=2 THEN G1=2:F1=0:POKE 3
    6876,250:POKE 36876,0:GOTO 400               <018>
380 GET A$                                       <080>
385 IF A$<>""THEN POKE 198,0:GOTO 800            <182>
400 POKE FA,114:SR=0:F1=0                        <094>
450 GOTO 300                                     <140>
480 REM*BEWEGUNG DER LEGIONEN*                   <100>
500 IF Z=0 THEN X=A(I)                           <236>
501 IF Z=1 THEN X=B(I)                           <015>
502 GOSUB 3000                                   <220>
505 W2=PEEK(X-22):IF W2<>32 AND W2<>34 AND
     W2<>115 THEN SW=1                           <106>
507 IF X<4118 THEN SW=0                          <081>
510 IF U2=1 AND BW(I)<2 AND SW=1 THEN POKE
     X,W1:X=X-22:BW(I)=BW(I)+1:W1=W2:GOTO
    600                                          <214>
515 W3=PEEK(X+22):IF W3<>32 AND W3<>34 AND
     W3<>115 THEN SW=2                           <222>
517 IF X>4557 THEN SW=0                          <063>
520 IF U4=1 AND BW(I)<2 AND SW=2 THEN POKE
     X,W1:X=X+22:BW(I)=BW(I)+1:W1=W3:GOTO
    600                                          <218>
521 W4=PEEK(X-1):IF W4<>32 AND W4<>34 AND
    W4<>115 THEN SW=3                            <035>
525 IF X=M OR X=M-22 OR X=M-44 OR X=M-66 O
    R X=M-88 OR X=M-198 OR X=M-220 OR X=M-
    242 THEN SW=0                                <110>
530 IF U1=1 AND BW(I)<2 AND SW=3 THEN POKE
     X,W1:X=X-1:BW(I)=BW(I)+1:W1=W4:GOTO 6
    00                                           <019>
532 W5=PEEK(X+1):IF W5<>32 AND W5<>34 AND
    W5<>115 THEN SW=4                            <230>
535 IF X=N OR X=N-22 OR X=N-44 OR X=N-66 O
    R X=N-88 OR X=N-110 OR X=N-220 OR X=N-
    242 OR X=N-264 THEN SW=0                     <189>
540 IF U3=1 AND BW(I)<2 AND SW=4 THEN POKE
     X,W1:X=X+1:BW(I)=BW(I)+1:W1=W5:GOTO 6
    00                                           <013>
600 IF Z=0 THEN POKE X,34:A(I)=X                 <198>
610 IF Z=1 THEN POKE X,115:B(I)=X                <205>
615 FOR Y=1 TO 5:IF B(I)=K(Y)THEN K(Y)=5:P
    OKE 36875,150:BE=BE+1:IF BE>=4 THEN BE
    =0:GOTO 2000                                 <022>
616 POKE 36875,0:NEXT Y                          <081>
620 SW=0:P1=0                                    <180>
630 IF SC=1 THEN G1=0:W1=0:POKE 36876,200:
    POKE 36876,0:GOTO 300                        <255>
700 GOTO 502                                     <216>
800 Z=Z+1:IF Z>1 THEN Z=0:ZT=ZT+1                <220>
801 IF ZT<>9 THEN 805                            <106>
802 FOR BO=1 TO 25:IF A(BO)>0 THEN K1(BO)=
    K1(BO)+100                                   <192>
803 IF B(BO)>0 THEN K2(BO)=K2(BO)+70             <087>
804 NEXT BO                                      <058>
805 IF ZT>=25 THEN 2500                          <105>
810 FOR T=1 TO 20                                <155>
815 POKE 36874,130+T*2                           <208>
819 FOR T2=1 TO 15                               <147>
820 IF A(T)+22=B(T2)OR A(T)-22=B(T2)OR A(T
    )-1=B(T2)OR A(T)+1=B(T2)THEN 840             <100>
830 GOTO 860                                     <148>
840 TR=INT(RND(1)*20)+1:K1(T)=K1(T)-TR:IF
    K1(T)<=0 THEN POKE A(T),116:A(T)=0           <070>
850 TL=INT(RND(1)*20)+1:K2(T2)=K2(T2)-TL         <164>
855 IF K2(T2)<=0 THEN POKE B(T2),140:B(T2)
    =0                                           <013>
860 NEXT T2                                      <049>
870 NEXT T                                       <024>
880 POKE 36874,0                                 <074>
1000 FOR T=1 TO 25:BW(T)=0:NEXT T                <148>
1010 G1=0:W1=0                                   <094>
1015 POKE 36869,192:PRINT"{CLR,HOME,6DOWN}
     WIR BEFINDEN UNS IM"                        <160>
1016 PRINT"{DOWN}JAHRE "JA+ZT" N. CHR."          <028>
1017 GOSUB 5000                                  <243>
1020 GOTO 110                                    <202>
2000 REM*SIEG DER GERMANEN*                      <003>
2010 POKE 36875,0:POKE 36869,192:POKE 3687
     9,25:PRINT"{CLR}"                           <001>
2100 PRINT"{HOME,RED,DOWN}DAS ROEMISCHE IM
     PERIUM{DOWN}LIEGT AM BODEN !"               <008>
2110 PRINT"{2DOWN}DIE GERMANEN HABEN{4SPAC
     E,DOWN}HABEN SIE WEGEN IHRER{SPACE,DO
     WN}FUEHRUNGSFAEHIGKEITEN"                   <017>
2120 PRINT"{DOWN}ZUM KOENIG AUF LEBENS-{DO
     WN}ZEIT ERNANNT."                           <066>
2130 PRINT"{DOWN}(DAS IST EINE GROSSE{2SPA
     CE,DOWN}EHRE,WENN MAN BEDENKT,{DOWN}D
     ASS BALD DIE HUNNEN"                        <219>
2140 PRINT"{DOWN}KOMMEN)!!"                      <237>
2150 GOTO 2640                                   <034>
2490 REM*SIEG DER ROEMER*                        <141>
2500 POKE 36869,192:POKE 36879,25:PRINT"{C
     LR,HOME,BLUE}":                             <227>
2600 PRINT"{DOWN}DAS ROEMISCHE IMPERIUM{DO
     WN}KONNTE DIESEN KLAEG-{2SPACE,DOWN}L
     ICHEN VERSUCH EINER"                        <252>
2610 PRINT"{DOWN}HORDE BARBAREN LAESSIG{DO
     WN}ABWEHREN."                               <110>
2620 PRINT"{DOWN}DEN GERMANEN FOLGENDEN{DO
     WN}TIP:"                                    <175>
2630 PRINT"{DOWN}VERSUCHT ES IN HUNDERT{DO
     WN}JAHREN NOCHEINMAL!"                      <031>
2640 PRINT"{2DOWN}NEUES SPIEL J/N ?"             <170>
2650 GET A$:IF A$="J"THEN ZT=0:RUN               <144>
2660 IF A$="N"THEN PRINT"{CLR,HOME,DOWN,RE
     D}IHR SEID DESERTEURE!!!":END               <210>
2670 GOTO 2650                                   <078>
3000 REM*JOYSTICKABFRAGE*                        <129>
3010 POKE U+3,127:U3=-((PEEK(U+1)AND 128)=
     0):POKE U+3,255                             <253>
3020 U1=-((PEEK(U)AND 16)=0):U4=-((PEEK(U)
     AND 8)=0):U2=-((PEEK(U)AND 4)=0)            <193>
3030 SC=-((PEEK(37137)AND 32)=0)                 <035>
3040 RETURN                                      <048>
4990 REM*BILDSCHIRMFARBEN*                       <247>
5000 PRINT"{3DOWN}WAEHLEN SIE DIE{7SPACE,D
     OWN}BILDSCHIRMFARBEN:"                      <111>
5001 PRINT"{DOWN,RVSON}F1{RVOFF,SPACE}= BI
     LDSCHIRMFARBE{2SPACE,DOWN,RVSON}F3{RV
     OFF,SPACE}= RAHMENFARBE "                   <023>
5002 PRINT"{DOWN,RVSON}S{RVOFF,2SPACE}= ST
     ART"                                        <012>
5005 POKE 36879,O1+O2:GET A$:IF A$="{F1}"T
     HEN O1=O1+16:IF O1>248 THEN O1=24           <019>
5010 IF A$="{F3}"THEN O2=O2+1:IF O2>7 THEN
      O2=0                                       <203>
5020 IF A$="S"THEN RETURN                        <110>
5030 GOTO 5005                                   <241>
6000 DATA 4192,4237,4282,4305,4306,4308,43
     10,4312                                     <064>
6010 DATA 4255,4301,4321,4420,4460,4428,45
     40,4361,4557                                <243>
6020 DATA 4368,4375,4355                         <128>
6030 DATA 4173,4194,4195,4196,4217,4218,42
     39,4240,4241                                <196>
6040 DATA 4242,4262,4263,4264,4265,4266          <174>
6050 DATA 4427,4544,4491,4437,4256               <151>
```


Listing 2. »Imperium Romanum«, Hauptprogramm (Schluß)

Spielhallengefühle mit dem VC 20

Wenn Sie schon immer ein witziges, schnelles Spiel mit flotter Musik für die Grundversion des VC 20 suchten, haben Sie jetzt allen Grund, in die Tasten zu greifen. »Tacco« läßt das Herz aller Spielernaturen höher schlagen.

Ordnung muß sein – auch in der Milchstraße. Das intergalaktische Straßensystem soll rot eingefärbt werden. Eine wichtige Aufgabe, die »Tacco«, einem rosa Raumschiff, zufällt. Der Spieler schlüpft in die Rolle von »Tacco« und muß unter Berücksichtigung einiger mordlüsterner Monster die Straßen anpinseln. Sind alle vier Eckfelder eingefärbt, flackert der Bildschirmrand für einige Sekunden in den wildesten Farben. In dieser Zeit kann »Tacco« sich an den Monstern revanchieren. Es darf sie verspeisen. Pro Monster werden satte 1000 Punkte gutgeschrieben. Nach Spielende wird der bestehende High Score eingeblendet.

»Tacco« ist ein reines Maschinencode-Programm, das die Grundversion des VC 20 bis zum letzten Bit in Anspruch nimmt. Durch den völligen Verzicht auf Basic-Teile wurde aber ein Spiel geschaffen, das zeigt, was man aus dem kleinen Commodore alles rausholen kann. Bevor Sie »Tacco« eintippen, müssen Sie das Eingabeprogramm (Listing 1) eingeben und speichern. Dazu benötigen Sie allerdings eine 3 KByte-Speichererweiterung. Lauffähig ist »Tacco« aber wohlgemerkt auf der Grundversion des VC 20. Sollten Sie über eine andere Speichererweiterung verfügen (8 KByte aufwärts), müssen Sie im Eingabeprogramm folgendes ändern: Die beiden POKEs in Zeile 99 entfallen. Bevor Sie das Eingabeprogramm einladen und starten, ist folgendes einzugeben:
`POKE 36866,PEEK (36866) OR 128: POKE648,30:POKE642,32: SYS64818`

Das Programm wird anschließend mit »RUN« gestartet und Sie können nun die vielen schönen Zahlen des »Tacco«-Listings 2 eintippen. Wie »Tacco« dann gespeichert wird, ist in den REM-Zeilen des Eingabeprogramms erläutert. Diese REM-Zeilen brauchen Sie natürlich nicht abtippen. Und noch eine Feinheit: Wer keine 3 KByte-Erweiterung benutzt, sollte statt »SYS 64802« lieber »SYS 64818« verwenden (betrifft Zeile 60 im Eingabeprogramm). Nach soviel Vorarbeit können Sie nun endlich das intergalaktische Straßensystem »erröten« lassen.

Hilfreiches beim Eintippen von Hexadezimalzahlen

Damit Sie nicht gleich die Lust beim Eintippen von langen Zahlenreihen verlieren, sollten Sie folgende Punkte beherzigen:

Vermeiden Sie, das ganze Programm auf einmal abzutippen. Die Konzentration verliert sich etwa auf halbem Wege. Die Eingabefehler werden häufiger und Sie müssen fehlerhafte Zeilen wiederholt eingeben.

```
1 REM                              ***************
  **                                          <138>
2 REM                              *     TACCO
  *                                           <018>
3 REM                              * EINGABEPROGR.
  *                                           <043>
4 REM                              ***************
  **                                          <141>
10 REM WENN SIE EINE 3K SPEICHERERWEITERUN
   G BESITZEN, KOENNEN SIE TACCO MIT          <178>
20 REM DIESEM PROGRAMM EINTIPPEN. DAS PROG
   RAMM GIBT AUCH DIE PRUEFSUMME AUS. DIE     <124>
30 REM JEWEILS LETZTE ZEILE KANN WIEDERHOL
   T EINGEGEBEN WERDEN.                       <249>
35 REM DIE REMARKZEILEN BITTE NICHT ABSCHR
   EIBEN!                                     <068>
40 REM TACCO WIRD MIT SYS5872 GESTARTET. V
   OR DEM ERSTEN START MUSS ES JEDOCH         <096>
50 REM ABGESPEICHERT SEIN. DIES GESCHIEHT
   WIEFOLGT:                                  <153>
60 REM SYS 64802
     [ SYSTEMRESET ]                          <068>
70 REM POKE43,1:POKE44,16:POKE4096,0
     [ BASICBEGINN INITIALISIEREN ]           <169>
75 REM NEW                                    <057>
76 REM 1 SYS5872
     [ DIESE ZEILE EINGEBEN ]                 <192>
80 REM POKE45,0:POKE46,30
     [ BASIC ENDE ]                           <102>
90 REM POKE51,20:POKE52,30
     [ FILENAME AUF BILDSCHIRM ]              <208>
91 REM BILDSCHIRM LOESCHEN UND MIT CURSOR
   IN DIE DRITTE ZEILE                        <212>
92 REM SAVE"TACCO"                            <169>
93 REM                                        <155>
94 REM                                        <156>
95 REM                                        <157>
99 PRINT"{CLR}":POKE 51,200:POKE 52,15        <015>
100 AD=4112                                   <082>
110 SU=0:FOR J=0 TO 19                        <178>
111 D=AD:GOSUB 400:PRINT H$;:INPUT H$         <249>
120 IF LEN(H$)<>2 THEN 111                    <140>
121 D=0:FOR I=0 TO 1                          <154>
122 T$=MID$(H$,2-I,1)                         <056>
123 IF T$<"0"OR T$>"9"AND T$<"A"OR T$>"F"T
    HEN 111                                   <051>
124 IF T$>"9"THEN D=D+(ASC(T$)-55)*16↑I:GO
    TO 126                                    <189>
125 D=D+(ASC(T$(-48)*16↑I                     <155>
126 NEXT                                      <136>
130 POKE AD,D:SU=SU+D:AD=AD+1:NEXT J          <207>
140 D=SU:GOSUB 400:PRINT"CHECKSUMME: {RVSON
    }"H$"{RVOFF,SPACE}OK?"                    <042>
150 GET F$:IF F$=""THEN 150                   <003>
160 IF F$="J"THEN 110                         <006>
170 IF F$="N"THEN AD=AD-20:GOTO 110           <045>
180 GOTO 150                                  <188>
400 H$="":FOR I=3 TO 0 STEP-1                 <030>
410 S=INT(D/16↑I):D=D-S*16↑I                  <030>
420 IF S>9 THEN H$=H$+CHR$(S+55):GOTO 440     <057>
430 H$=H$+CHR$(S+48)                          <209>
440 NEXT I:RETURN                             <130>
```


Listing 1. Eingabeprogramm zu »Tacco«

Machen Sie des öfteren längere Pausen, in denen Sie sich mit anderen Dingen beschäftigen. Am besten ist es, an die frische Luft zu gehen und Sauerstoff zu tanken, bevor Sie den Endspurt starten.

Suchen Sie sich möglichst einen ruhigen Raum, in dem Sie wenig gestört werden. Jede Störung vermindert Ihre Konzentration erheblich.

Vergessen Sie keinesfalls vor dem ersten Test das Programm zu speichern. Bei einem Programmabsturz könnte Ihnen sonst der gesamte Speicherinhalt verlorengehen und die Arbeit wäre umsonst gewesen.

Wollen Sie sich den Aufwand des Eintippens ersparen, wäre es natürlich auch möglich, sich über unseren Programmservice eine Programm-Diskette oder -Kassette zum VC 20 anzufordern.

(Richard Weisang/Frank Ammann/do)

```
1010: D8 A0 00 84 45 A9 80 8D 67 03 85 FC A9 01 85 FE 20 C4 1A A9  ;09B6
1024: FF 8D 05 90 20 5F E5 A9 08 85 01 8D 0F 90 A9 1E 85 02 A9 07  ;07E6
1038: A0 FF 99 00 96 88 D0 FA A0 FF 99 FF 96 88 D0 FA A2 00 BD 5B  ;0CF9
104C: 1A C9 EA D0 12 E8 18 A5 01 7D 5B 1A 85 01 A5 02 69 00 85 02  ;0764
1060: 4C 72 10 91 01 18 A5 01 69 01 85 01 A5 02 69 00 85 02 E8 E0  ;066D
1074: 59 D0 D3 A2 30 A0 30 A9 00 85 08 20 B3 1A A5 CB C9 27 D0 07  ;08F8
1088: E8 E0 3A D0 02 A2 30 C9 2F D0 07 C8 C0 38 D0 02 A0 30 8E D5  ;0A3A
109C: 1E 8C 01 1F 48 A9 AA 8D 50 03 68 20 44 1A C9 37 F0 26 AD 49  ;0737
10B0: 03 C9 07 D0 08 A9 02 8D 49 03 4C C2 10 A9 07 8D 49 03 86 FB  ;0757
10C4: A2 FF 9D 00 96 9D FF 96 CA D0 F7 A6 FB 4C 7B 10 8C 48 03 EA  ;0BD0
10D8: A0 00 20 50 17 A2 15 BD 2D 1A 9D 00 1E CA 10 F7 A2 15 BD 17  ;06F9
10EC: 1A 9D 16 1E CA 10 F7 A9 2C 85 01 A9 1E 85 02 A9 1E A2 14 A0  ;0782
1100: 15 91 01 88 88 88 10 F9 18 A5 01 69 16 85 01 A5 02 69 00 85  ;06A0
1114: 02 A9 1E CA D0 E5 A2 58 86 01 A2 1E 86 02 A2 07 A0 15 B9 01  ;0829
1128: 1A 91 01 88 10 F8 18 A5 01 69 42 85 01 A5 02 69 00 85 02 CA  ;068C
113C: D0 E6 A2 15 BD EB 19 9D E4 1F CA 10 F7 A9 02 A2 04 9D 08 96  ;0A2B
1150: CA 10 FA A2 03 9D ED 97 CA 10 FA A2 05 A9 05 9D 10 96 CA 10  ;09E0
1164: FA A2 03 A9 04 9D 00 96 CA 10 FA 20 62 19 20 C2 17 A0 00 38  ;07BF
1178: AD 48 03 E9 2F AA A9 26 99 2C 1E A9 06 99 2C 96 C8 C8 C8 98  ;0966
118C: 48 8A 48 A2 80 A0 FF 8E 0B 90 8E 0A 90 88 D0 F7 E8 D0 F2 68  ;0B8D
11A0: AA 68 A8 CA D0 D8 8E 0B 90 8E 0A 90 A9 18 8D EE 1F A9 04 8D  ;0A12
11B4: EE 97 EA A9 44 8D 70 03 A2 23 A9 1E 9D 81 03 9D 83 03 AD 70  ;0949
11C8: 03 38 E9 03 8D 70 03 9D 80 03 A9 16 9D 82 03 CA CA CA CA CA  ;091A
11DC: 10 E0 AD 67 03 A2 00 AD 48 03 38 E9 2F A8 88 E8 E8 E8 E8 E8  ;0AA9
11F0: 88 D0 F8 20 50 18 EA BD 80 03 85 2D BD 81 03 85 2E BD 83 03  ;08EB
1204: A0 00 91 2D A5 2E 18 69 78 85 2E BD 84 03 91 2D A5 2E 38 E9  ;07D3
1218: 78 85 2E 20 00 19 B1 2D 9D 83 03 C9 26 D0 14 A8 BD 82 03 49  ;076B
122C: FF 18 69 01 9D 82 03 98 4C F7 11 EA EA A0 00 C9 1A F0 0C C9  ;09AB
1240: 18 F0 08 C9 11 F0 04 C9 16 D0 03 4C 22 15 C9 25 D0 16 A8 BD  ;084C
1254: 82 03 C9 16 D0 08 A9 FF 9D 82 03 4C 67 12 A9 EA 9D 82 03 98  ;0918
1268: C9 1C D0 16 A8 BD 82 03 C9 01 D0 08 A9 16 9D 82 03 4C 81 12  ;0817
127C: A9 FF 9D 82 03 98 C9 1B D0 16 A8 BD 82 03 C9 16 D0 08 A9 01  ;0977
1290: 9D 82 03 4C 9B 12 A9 EA 9D 82 03 98 C9 1D D0 16 A8 BD 82 03  ;091E
12A4: C9 FF D0 08 A9 16 9D 82 03 4C B5 12 A9 01 9D 82 03 98 EA C9  ;09AB
12B8: 00 D0 66 A8 BD 82 03 C9 01 D0 12 AD 14 91 10 05 A9 EA 4C CF  ;08E1
12CC: 12 A9 16 9D 82 03 4C 1E 13 BD 82 03 C9 FF D0 12 AD 14 91 10  ;07BE
12E0: 05 A9 EA 4C E8 12 A9 16 9D 82 03 4C 1D 13 BD 82 03 C9 EA D0  ;0900
12F4: 12 AD 14 91 10 05 A9 FF 4C 01 13 A9 01 9D 82 03 4C 1D 13 BD  ;0686
1308: 82 03 C9 16 D0 0F AD 14 91 10 05 A9 FF 4C 1A 13 A9 01 9D 82  ;0794
131C: 03 98 EA EA EA C9 0A D0 20 A8 BD 82 03 C9 FF D0 12 AD 14 91  ;0B02
1330: 10 05 A9 EA 4C 39 13 A9 16 9D 82 03 4C 44 13 A9 01 9D 82 03  ;0690
1344: 98 C9 0B D0 20 A8 BD 82 03 C9 EA D0 12 AD 14 91 10 05 A9 FF  ;09EA
1358: 4C 5D 13 A9 01 9D 82 03 4C 68 13 A9 16 9D 82 03 98 C9 21 D0  ;0782
136C: 20 A8 BD 82 03 C9 16 D0 12 AD 14 91 10 05 A9 FF 4C 81 13 A9  ;0863
1380: 01 9D 82 03 4C 8C 13 A9 EA 9D 82 03 98 C9 28 D0 20 A8 BD 82  ;0923
1394: 03 C9 01 D0 12 AD 14 91 10 05 A9 EA 4C A5 13 A9 16 9D 82 03  ;078E
13A8: 4C B0 13 A9 FF 9D 82 03 98 EA EA EA A0 00 A9 26 91 2D 18 A5  ;0A19
13BC: 2E 69 78 85 2E A9 03 EA 91 2D 38 A5 2E E9 78 85 2E CA CA CA  ;0993
13D0: CA CA 30 03 4C F7 11 20 44 1A EA A0 00 B1 01 8D C0 03 8C AA  ;085B
13E4: 03 EA EA EA EA EA 20 8D 19 AD 60 03 C9 00 D0 08 8D A9 03 A9  ;09EE
13F8: 01 8D AA 03 AD 61 03 C9 00 D0 08 8D A9 03 A9 16 8D AA 03 AD  ;07CC
140C: 62 03 C9 00 D0 08 A9 FF 8D A9 03 8D AA 03 AD 63 03 C9 00 D0  ;08CD
1420: 0A A9 FF 8D A9 03 A9 EA 8D AA 03 AD AA 03 C9 00 D0 03 4C 49  ;0943
1434: 1D AD AB 03 91 01 AD AC 03 91 33 A5 01 18 6D AA 03 85 01 A5  ;072D
1448: 02 6D A9 03 85 02 A5 33 18 6D AA 03 85 33 A5 34 6D A9 03 85  ;06DB
145C: 34 B1 01 C9 26 D0 03 4C 66 14 A2 0B BD D3 18 D1 01 D0 03 4C  ;07B4
1470: 9F 14 CA 10 F3 A5 01 38 ED AA 03 85 01 A5 02 ED A9 03 85 02  ;0845
1484: A5 33 38 ED AA 03 85 33 A5 34 ED A9 03 85 34 AD C0 03 91 01  ;088F
1498: A9 04 91 33 4C 49 1D 4C 00 17 EA AD AC 03 8D BA 03 EA EA EA  ;08D4
14AC: EA EA EA EA B1 33 29 0F 8D AC 03 B1 01 8D AB 03 AD AA 03 C9  ;0A10
```

Listing 2. Maschinencode-Programm »Tacco«

```
14C0: 01 D0 02 A2 18 C9 FF D0 02 A2 16 C9 16 D0 02 A2 11 C9 EA D0  ;09C6
14D4: 02 A2 1A 8A 91 01 A9 04 91 33 EA EA 4C 1C 16 AD BA 03 29 0F  ;073F
14E8: C9 02 D0 30 AD AC 03 29 0F C9 07 D0 27 A5 08 C9 12 D0 03 4C  ;07CD
14FC: 1C 15 A6 08 A5 33 9D C4 03 A5 34 9D C5 03 E6 08 E6 08 A5 08  ;07E2
1510: C9 44 F0 08 A9 02 8D AC 03 4C 49 1D 20 B3 16 4C 49 1D 20 B3  ;070C
1524: 16 38 AD 48 03 E9 30 A2 00 A8 E8 E8 E8 E8 E8 88 D0 F8 A0 00  ;0B21
1538: BD 80 03 85 2F BD 81 03 18 69 78 85 30 BD 84 03 A0 00 91 2F  ;0787
154C: CA CA CA CA CA 10 E5 A5 FE F0 35 A9 10 A2 F0 8E 0F 90 8E 0C  ;0BC1
1560: 90 A0 FF 88 D0 FD CA 30 F2 38 E9 01 D0 EB A9 08 8D 0F 90 A2  ;0BCC
1574: 03 BD 00 1E C9 1A F0 06 CA 10 F6 4C BE 15 A9 20 9D 00 1E A9  ;07D3
1588: FF 4C AA 15 A9 C8 A0 F0 A2 50 8C 0A 90 CA 8E 0B 90 30 FA 88  ;0AC8
159C: 30 F2 A0 05 48 20 B0 18 68 38 E9 01 D0 E4 A5 02 18 69 78 85  ;085A
15B0: 02 AD AC 03 A0 00 91 01 20 62 19 4C E8 10 A2 09 BD DE 18 9D  ;076A
15C4: 0E 1F A9 01 9D 0E 97 CA 10 F2 AE 0E 90 20 44 1A 20 44 1A 8E  ;06BB
15D8: 0E 90 CA 10 F4 A2 00 BD 10 1E DD F3 03 30 05 F0 05 4C D7 17  ;0830
15EC: D0 05 E8 E0 06 D0 EC 4C E2 17 A2 05 A9 00 9D F3 03 CA 10 FA  ;0A5B
1600: 60 00 00 00 00 00 AD AC 03 CD BA 03 D0 09 A9 00 85 08 A9 06  ;0604
1614: 8D F0 03 4C E3 14 EA EA A5 0A C9 00 F0 13 AD AC 03 C9 02 D0  ;0A09
1628: 0C A9 02 8D AC 03 A9 02 8D BA 03 EA EA A9 07 85 09 EA A5 08  ;0891
163C: C9 02 D0 04 4C 6B 16 00 AD AA 03 49 FF AA E8 8A CD F0 03 D0  ;09BA
1650: 0D A9 AA 8D F0 03 A9 00 EA EA EA 4C 1C 15 AD AA 03 C9 00 F0  ;09D7
1664: 03 8D F0 03 4C 90 16 AD AA 03 49 FF AA E8 8A CD F0 03 D0 18  ;09DB
1678: A9 02 8D BA 03 8D F0 03 A9 02 8D AC 03 A9 02 85 08 A9 02 85  ;07C4
168C: 09 4C 1C 15 A5 08 C9 00 F0 18 AD AC 03 CD BA 03 D0 10 A0 19  ;0783
16A0: E6 0F A5 0F C9 31 D0 03 4C 61 17 20 91 18 A0 00 4C 06 16 A0  ;06AB
16B4: 00 A6 08 F0 36 CA CA BD C4 03 85 3F 85 31 BD C5 03 85 40 85  ;0935
16C8: 32 B1 3F 29 0F C9 04 F0 04 A9 07 91 3F 20 0D 18 CA CA 10 DF  ;0763
16DC: A9 00 85 08 A5 09 8D AC 03 A9 03 85 0A 8C 0A 90 8C 0B 90 60  ;0708
16F0: A2 10 A9 20 9D FA 1F CA 10 FA 20 F6 15 4C 10 10 B1 01 AE AA  ;08A6
1704: 03 E0 01 D0 16 C9 1B F0 0F C9 1D F0 0B C9 1E F0 07 C9 0A F0  ;092F
1718: 03 4C 1F 17 4C 75 14 E0 FF D0 16 C9 25 F0 0F C9 1C F0 0B C9  ;08B5
172C: 1E F0 07 C9 28 F0 03 4C 39 17 4C 75 14 A5 02 C9 20 F0 0A C9  ;07BD
1740: 1E F0 09 A5 01 C9 FB 90 03 4C 75 14 4C A2 14 00 20 5F E5 A2  ;07F1
1754: FF A9 07 9D 15 96 9D 14 97 CA D0 F5 60 A9 25 85 0E A2 FF A0  ;0AD0
1768: 08 84 0D A0 08 98 05 0D EA 8D 0F 90 20 B0 17 EA 88 C0 08 D0  ;07F2
177C: F0 CA D0 E7 C6 0E A5 0E 10 DF AD 48 03 C9 37 F0 03 EE 48 03  ;0A0B
1790: A0 00 A9 2C 85 01 A9 1E 85 02 20 53 17 A9 08 8D 0F 90 A9 00  ;0659
17A4: 85 0F 85 08 A9 80 85 FC 4C E8 10 EA A8 8A 09 A0 8D 0A 90 09  ;0904
17B8: 22 8D 0B 90 09 33 8D 0C 90 60 A2 23 A0 15 B9 2C 96 9D 84 03  ;0728
17CC: 88 88 88 CA CA CA CA CA 10 F0 60 A2 05 BD 10 1E 9D F3 03 CA  ;0AD9
17E0: 10 F7 A2 05 BD F3 03 9D 00 1E CA 10 F7 A2 05 A9 04 9D 00 96  ;0874
17F4: CA 10 FA A5 CB C9 40 F0 FA 4C 11 10 00 00 8E 66 03 A2 00 A1  ;08DE
1808: AC AE 66 03 60 8A 48 98 AA 38 A5 32 E9 78 85 32 BD 80 03 C5  ;0963
181C: 31 D0 0C BD 81 03 C5 32 D0 05 A9 07 9D 84 03 E8 E8 E8 E8 E8  ;0A76
1830: E0 28 D0 E4 68 AA 09 81 8D 0A 90 09 AA 8D 0B 90 A0 FF 88 D0  ;0A51
1844: FD A0 FF 88 D0 FD A0 00 60 00 00 00 EA 8A 48 98 48 A5 45 C9  ;0A40
1858: 00 F0 24 A2 09 A0 0D 20 85 18 EA 98 09 DD 8D 0A 90 88 D0 F3  ;0903
186C: 8A 09 F0 8D 0B 90 CA D0 E8 A9 00 8D 0A 90 8D 0B 90 C6 45 68  ;0998
1880: A8 68 AA 60 00 98 48 A0 22 EA EA 88 D0 FB 68 A8 60 A2 06 BD  ;0AB8
1894: 31 1B 85 AC BD 32 1B 85 AD 20 02 18 EA 29 0F C9 02 D0 09 CA  ;0783
18A8: CA 10 E8 A9 00 85 FE EA A9 02 85 45 18 A2 05 88 10 01 60 BD  ;08C2
18BC: 10 1E C9 39 F0 08 69 01 9D 10 1E 4C B5 18 A9 30 9D 10 1E CA  ;06E4
18D0: 4C BB 18 25 1C 1B 1D 1F 1E 00 0A 0B 21 28 07 01 0D 05 20 20  ;028D
18E4: 0F 15 05 12 A9 90 8D 50 03 A5 FB 38 E9 2F AA 38 AD 50 03 E9  ;080F
18F8: 09 8D 50 03 CA 10 F4 60 A9 00 8D 72 03 BD 82 03 C9 FF F0 04  ;08C0
190C: C9 EA D0 05 A9 FF 8D 72 03 18 A5 2D 7D 82 03 85 2D A5 2E 6D  ;0910
1920: 72 03 85 2E A9 00 8D 72 03 BD 82 03 C9 FF F0 04 C9 EA D0 05  ;0959
1934: A9 FF 8D 72 03 18 BD 80 03 7D 82 03 9D 80 03 BD 81 03 6D 72  ;0844
1948: 03 9D 81 03 A0 00 A5 2E 18 69 78 85 2E B1 2D 9D 84 03 A5 2E  ;0718
195C: 38 E9 78 85 2E 60 A9 1F 85 02 A9 EE 85 01 A9 97 85 34 A9 EE  ;09A8
```

Listing 2. Maschinencode-Programm »Tacco« (Fortsetzung)

```
1970: 85 33 20 E8 18 A9 02 8D AC 03 A9 1F 8D AB 03 60 A5 FE F0 05  ;08BA
1984: A9 08 8D 0F 90 4C E1 11 00 A9 7F 8D 22 91 AD 20 91 29 80 8D  ;0817
1998: 60 03 A9 FF 8D 22 91 AD 11 91 29 08 8D 61 03 AD 11 91 29 10  ;0744
19AC: 8D 62 03 AD 11 91 29 04 8D 63 03 A6 CB E0 0D D0 08 AD 63 03  ;07AA
19C0: 29 00 8D 63 03 E0 15 D0 08 AD 62 03 29 00 8D 62 03 E0 25 D0  ;06EB
19D4: 08 AD 61 03 29 00 8D 61 03 E0 16 D0 08 AD 60 03 29 00 8D 60  ;0627
19E8: 03 60 00 1B 1F 1F 21 1F 1F 21 1F 1F 21 1F 1F 21 1F 1F 21 1F  ;0278
19FC: 1F 21 1F 1F 25 0A 1F 1F 00 1F 1F 00 1F 1F 00 1F 1F 00 1F 1F  ;01E3
1A10: 00 1F 1F 00 1F 1F 28 1D 1F 1F 0B 1F 1F 0B 1F 1F 0B 1F 1F 0B  ;01E5
1A24: 1F 1F 0B 1F 1F 0B 1F 1F 1C 1A 1A 1A 20 20 20 20 20 14 01 03  ;01F2
1A38: 03 0F 20 20 20 30 30 30 30 30 30 30 48 8A 48 98 48 AE 50 03  ;04BD
1A4C: A0 EE 88 D0 FD CA D0 F8 68 A8 68 AA 68 60 00 14 01 03 03 0F  ;0989
1A60: EA 24 23 03 24 20 31 39 38 33 20 02 19 EA 1D 06 27 01 0D 0D  ;03D7
1A74: 01 0E 0E 20 22 20 12 27 17 05 09 13 01 0E 07 EA 5D 06 31 20  ;02A4
1A88: 20 13 10 05 05 04 20 20 20 EA 20 06 33 20 20 0C 05 15 05 0C  ;026B
1A9C: 20 20 20 EA 77 10 12 05 13 13 20 06 35 20 14 0F 20 13 14 01  ;02F4
1AB0: 12 14 EA 8A 48 A6 07 E0 0C F0 04 E8 8E 0E 90 86 07 68 AA 60  ;0882
1AC4: A9 00 85 0F A2 00 8E 46 03 8E 41 03 8E 45 03 A9 E0 8D 14 03  ;068B
1AD8: A9 1A 8D 15 03 86 07 60 A9 F9 8D 18 03 A9 1B 8D 19 03 EA EA  ;07E0
1AEC: EA EA EA EA EA EA AE 46 03 EE 41 03 AD 41 03 DD 47 1B 30 22  ;0A27
1B00: A9 FD 8D 0D 90 EE 45 03 AC 45 03 C0 03 D0 1E A9 00 8D 41 03  ;0825
1B14: 8D 45 03 E8 E8 E0 B0 D0 02 A2 00 8E 46 03 BD 46 1B 8D 0C 90  ;08C7
1B28: A9 00 8D 0D 90 4C BF EA 60 16 96 E4 97 F9 97 2B 96 EA EA EA  ;0B5E
1B3C: EA EA EA EA EA EA EA BF EA 00 D7 12 D2 0C C3 06 C3 12 C3 0C  ;0C43
1B50: C3 06 C3 12 C3 0C D7 06 D2 0C C3 06 D2 0C D9 06 E1 12 DD 0C  ;088A
1B64: D2 06 D2 12 D2 0C D2 06 D2 12 D2 12 D2 12 D2 12 E1 12 DD 0C  ;08DE
1B78: D2 06 D2 12 D2 12 D2 12 D2 12 D7 0C D9 06 DD 0C D9 06 E1 12  ;08E5
1B8C: E1 12 E1 12 E1 12 E1 12 E1 12 E1 12 E1 12 D7 12 D2 0C C3 06  ;0935
1BA0: C3 12 C3 0C C3 06 C3 12 C3 0C D7 06 D2 0C C3 06 D2 0C D9 06  ;0852
1BB4: E3 12 E3 12 D9 12 D9 0C D9 06 D2 12 D2 12 C7 12 C7 12 E1 0C  ;0900
1BC8: DD 06 DD 0C D2 06 E1 0C DD 06 DD 0C D2 06 E1 0C DD 06 DD 0C  ;08EE
1BDC: CD 06 E1 0C DD 06 DD 0C CD 06 D9 12 D9 12 D9 12 D9 12 D7 12  ;08F4
1BF0: D7 12 D7 12 D7 12 01 06 00 58 A2 00 9A 4C 11 10 24 24 E7 00  ;05F2
1C04: 00 E7 24 24 30 78 CC C6 FE FE C6 C6 FC FE C2 FE FE C2 FE FC  ;0E65
1C18: 3E 7E E0 C0 C0 C0 7E 3E FC FE C6 C6 C6 C6 FE FC FE FE C0 F0  ;0F50
1C2C: F0 C0 FE FE FE FE C0 F0 F0 C0 C0 C0 7C FE C6 C0 DE C6 FE 7C  ;10A6
1C40: C6 C6 C6 FE FE C6 C6 C6 3C 3C 18 18 18 18 3C 3C 24 24 27 20  ;087F
1C54: 20 27 24 24 00 00 FF 00 00 E7 24 24 C0 C0 C0 C0 C0 C0 FE FE  ;0939
1C68: C6 EE FE D6 C6 C6 C6 C6 C6 E6 F6 F6 DE DE CE C6 7C FE C6 C6  ;108E
1C7C: C6 C6 FE 7C FC FE C6 FE FC C0 C0 C0 C3 FF 7E 7E 3C 3C 18 18  ;0D66
1C90: FC FE C6 FE FC F0 D8 CC 7C FE C6 70 1C C6 FE 7C 7E 7E 18 18  ;0D86
1CA4: 18 18 18 18 C6 C6 C6 C6 6C 6C 38 38 03 0F 3E FE FE 3E 0F 03  ;075C
1CB8: C6 C6 C6 C6 D6 FE EE C6 C0 F0 7C 7F 7F 7C F0 C0 66 66 66 66  ;0D8E
1CCC: 3C 18 18 18 18 18 3C 3C 7E.7E FF C3 24 24 27 20 20 3F 00 00  ;04D8
1CE0: 00 00 FC 04 04 E4 24 24 00 00 3F 20 20 27 24 24 24 24 24 24  ;03AE
1CF4: 24 24 24 24 00 00 FF 00 00 FF 00 00 00 00 00 00 00 00 00 00  ;028E
1D08: 24 24 E7 00 00 FF 00 00 7C 6C 78 30 6E C6 7C 3A 06 0C 18 18  ;05EA
1D1C: 18 18 0C 06 60 30 18 18 18 18 30 60 24 24 E4 04 04 FC 00 00  ;03F2
1D30: 3C 42 A5 81 C3 BD 42 3C 00 00 00 00 00 00 18 18 24 24 E4 04  ;0502
1D44: 04 E4 24 24 EA A5 FE D0 29 A5 FC C9 FF F0 1F A5 4D C9 01 D0  ;0BBA
1D58: 0A A9 0A E6 4D 8D 0F 90 4C 6C 1D A9 01 85 4D A9 0C 8D 0F 90  ;074E
1D6C: C6 FC A5 FC D0 06 A9 FF 85 FC 85 FE 4C D0 1D EA EA EA EA EA  ;0EB0
1D80: 7C FE C6 CE D6 E6 FE 7C 0C 1C 3C 3C 2C 0C 1E 1E 3C 7E C6 1C  ;08F4
1D94: 38 60 FE FE 3C FE C6 1C 1C C6 FE 7C C0 C0 D8 D8 FE FE 18 18  ;0C68
1DA8: FE FE C0 FC FE 06 FE 7C 1E 3E 60 FC FE C6 FE 7C FE FE 06 06  ;0D34
1DBC: 3C 3C 30 60 7C FE C6 7C 7C C6 FE 7C 7C FE C6 FE 7E 06 0C 18  ;0A66
1DD0: AD 67 03 C9 AA F0 26 AD 10 1E C9 30 D0 1F AD 11 1E C9 31 D0  ;0909
1DE4: 18 A2 FF E8 BD 00 1E C9 20 D0 F8 A9 1A 9D 00 1E A9 AA 8D 67  ;09F2
1DF8: 03 A9 03 85 45 4C 80 19 B2 00 FF 00 FF 00 FF 00 FF 00 FF 00  ;080B
1E0C: FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00  ;09F6
```

Listing 2. Maschinencode-Programm »Tacco« (Schluß)


PROTEXT
TOP
ASS

C64-Programme für C16 und VC20

In vielen Zeitschriften und Büchern findet man Programme für den C64, während es um Software für C16 und VC 20 schon schlechter bestellt ist. Hier sind einige Tips, wie man C64-Programme für den eigenen Computer umschreiben kann.

Leider wird man nur sehr selten Programme für den C64 finden, die ohne Änderungen auf dem VC 20 oder C16 laufen. Zwar sind Betriebssystem und Basic des VC 20 und des C64 praktisch identisch, aber selbst hier stimmen die sonstigen Hardware-Eigenschaften überhaupt nicht überein. Noch schlimmer wird es beim C16, bei dem es verschiedene Speicher-Bänke gibt für den dem Anwender zur Verfügung stehenden Speicher sowie für Betriebssystem und Basic. Diese Speicherbänke werden durch ein recht kompliziertes Bank-Switching-Verfahren, ähnlich wie beim C128, verwaltet.

Immer dann, wenn zum Beispiel die grafischen oder musikalischen Fähigkeiten des C64 angesprochen werden, geschieht dies durch POKE-Befehle, mit denen Video- und Sound-Chip direkt gesteuert werden. Aufgrund der unterschiedlichen Hardware von C64, VC 20 und C16, die zudem auch noch völlig unterschiedliche Adreßbereiche belegt, ist ein C64-Programm, das irgendwelche POKE-Befehle enthält, in der Regel weder auf dem VC 20 noch auf dem C16 lauffähig. Tabelle 1 gibt einen ersten Überblick über wichtige Speicheradressen bei allen drei Computern.

Wir wollen uns im folgenden damit beschäftigen, inwiefern man dennoch C64-Programme auf den VC 20 und den C16 übertragen kann. Dabei beschränken wir uns auf Basic-Programme, denn das Übertragen von Maschinencode-Programmen ist noch um einiges schwieriger und erfordert gute Kenntnisse und Erfahrung in Maschinensprache.

Generell kann man sagen, daß alle C64-Programme, die die fünf Befehle POKE, PEEK, SYS, USR und WAIT nicht enthalten, ohne jegliche Änderung sowohl auf dem VC 20 als auch auf dem C16 lauffähig sind – sofern sie natürlich nicht die Kapazität des bei beiden Computern in der Grundversion doch arg beschränkten Speichers überschreiten.

Beim VC 20 tritt noch das spezielle Problem des anderen Bildschirmformats auf. Während C64 und C16 auf dem Bildschirm 25 Zeilen zu je 40 Zeichen darstellen, sind es beim VC 20 nur 23 Zeilen zu je 22 Zeichen. Dadurch gerät bei einfacher Übernahme der C64-Vorlage beim VC 20 das ganze Bildschirm-Layout aus den Fugen. Eine Anpassung der PRINT-Anweisungen im Programm an die veränderten Verhältnisse ist also in fast allen Fällen erforderlich, dürfte aber in der Regel auch kein großes Problem sein. Bei der Anpassung von C64-Programmen an den VC 20 wird man des öfteren zusätzliche Print-Befehle einfügen, da eine Bildschirmzeile beim VC 20 nur etwa halb soviel Zeichen aufnehmen kann, wie das beim C64 der Fall ist.

Ernste Schwierigkeiten kann es möglicherweise bei der Darstellung von Tabellen geben. Eine sechsspaltige Zahlentabelle zum Beispiel läßt sich beim C64 und C16 ganz gut darstellen, beim VC 20 wird man bei derartigen Versuchen unangenehm an die arg begrenzte Zeilenbreite erinnert. In solchen Fällen kann man versuchen, weniger interessante Spalten der Tabelle einfach wegzulassen oder die Tabelle in zwei Teilen auszugeben. Wenn das nicht erwünscht ist, hilft nur noch der CMD-Befehl, um die Ausgabe der entsprechenden Tabelle auf den Drucker umzuleiten.

Was nicht geht

Leider gibt es viele Programme für den C64, die man nicht durch Änderung einiger POKE-Adressen und kleinerer Korrekturen am Bildschirmaufbau übernehmen kann. Dazu gehören beispielsweise die meisten Spiele, jedenfalls soweit es sich um Video-Spiele handelt. Der Grund hierfür sind die beim C64 vorhandenen acht Hardware-Sprites, die sich weder auf dem VC 20 noch auf dem C16 reproduzieren lassen. Sprites sind kleine, bewegliche Grafikobjekte, die vom Video-Chip des C64 in den normalen Bildschirm eingeblendet werden können. Sprites sind nicht zu verwechseln mit den beim C16 vorhandenen Shapes. Shapes sind einfach rechteckige Ausschnitte aus der hochauflösenden Grafik des C16, die mittels spezieller Basic-Befehle vom Bildschirm gelesen und auf ihn geschrieben werden können. Shapes werden also rein softwaremäßig erzeugt, während die Sprites des C64 von der Hardware, nämlich vom Video-Chip, kontrolliert werden. Der Programmierer muß, nachdem er die Form eines Sprites einmal festgelegt hat, nur noch die Sprite-Position in spezielle dafür bestimmte Register des Video-Chips POKEn, um Bewegungsabläufe zu erzeugen. In der Regel werden Sprites beim C64 auch nicht im Grafik-Modus verwendet, sondern im normalen Textmodus, was bei den Shapes des C16 zum Beispiel auch nicht möglich ist.

Ebenfalls nicht verwenden kann man Musik-Programme für den C64. Im Gegensatz zu VC 20 und C16 verfügt der C64 über einen echten Synthesizer-Baustein, mit dem sich eine ungeheure Fülle von Klangeffekten realisieren läßt. Dies alles führt dazu, daß sich leider ein Großteil aller C64-Programme, darunter die meisten Spiele, praktisch überhaupt nicht an den VC 20/C16 anpassen lassen – es sei denn, man würde diese Programme völlig neu entwickeln. Nach diesen eher etwas ernüchternden Feststellungen wollen wir uns jetzt den Programmen zuwenden, die sich mit vertretbarem Aufwand umschreiben lassen. Das sind jene Basic-Programme, die keine Maschinensprache-Routinen enthalten und nur normale Textausgabe (allenfalls noch die Blockgrafik) verwenden, also keine »Tips&Tricks-Routinen«

Eine Grundregel

Beim Umschreiben eines C64-Programms auf den C16 oder VC 20 sollte man eine ganz grundlegende Regel beachten: Niemals einen der bereits genannten »gefährlichen« Befehle POKE, SYS oder USR beim Abtippen des Listings so ohne weiteres übernehmen, sondern diese Befehle grundsätzlich nur übernehmen (meist in sinnvoll abgeänderter Form), wenn man sich über ihre Wirkung im klaren ist. Diese drei Befehle sind als einzige imstande, Speicherinhalte ungewollt zu verändern oder unkontrollierte Maschinensprache-Aufrufe zu bewirken. Hat man Bedenken, weil man sich nicht genau darüber im klaren ist, ob einer dieser Befehle schädlich ist oder nicht, dann empfiehlt es sich, den Befehl in eine PRINT-Anweisung umzuwandeln. Damit ist gemeint, den Befehl einfach in Anführungszeichen zu setzen und eine PRINT-Anweisung davor zu schreiben. Beispiel: Aus der Zeile

```
10 FOR I=49152 TO 49233 : READ X: POKE I,X : NEXT
```
wird durch diese Methode die folgende Zeile
```
10 FOR I=49152 TO 49233 : READ X: PRINT "POKE"
; I; X :NEXT
```
Dadurch werden die POKE-Befehle innerhalb der FOR-NEXT-Schleife, die beim C64 durchaus sinnvoll sind, bei anderen Computern aber bestenfalls nutzlos, nicht ausgeführt, sondern nur am Bildschirm protokolliert. Bei einem Probelauf des Programms bekommt man so eher einen Eindruck, was da eigentlich vor sich geht, und kann möglicherweise entsprechende Schlüsse für die Anpassung einer solchen Routine an den eigenen Computer ziehen.

Natürlich kann ein solches Programm nicht richtig laufen, aber es kann – und das ist die positive Seite – auf gar keinen Fall durch unkontrollierte POKE- oder SYS-Befehle »abstürzen«. Weniger problematisch sind die Befehle PEEK und WAIT, da hierdurch nichts unkontrolliert verändert werden kann. Gleichwohl sind auch diese beiden Basic-Befehle nicht unproblematisch. In den seltensten Fällen können die beim C64 verwendeten Adressen für diese Befehle auch beim VC 20/C16 benutzt werden. Daher empfiehlt sich hier der Einbau einer zusätzlichen PRINT-Anweisung unmittelbar vor dem WAIT-Befehl (etwa in der Art »PRINT "Zeile 255, WAIT-BEFEHL !"«) oder unmittelbar nach PEEK. Zum Beispiel sollte man die Zeile
```
10 IF PEEK(X)<>32 THEN 10
```
ändern in
```
10 IF PEEK(X) <>32 THEN PRINT"X=",X,"PEEK(X) ="
,PEEK(X):GOTO 10
```
Hierdurch hat man die Gewähr, beim Programmlauf ständig den Überblick zu behalten, was sich da gerade abspielt. Natürlich muß man sich nach dem Probelauf überlegen, wie man die POKE-Befehle (und andere) sinnvoll für seinen Computer abändert. Hierfür kann es naturgemäß kein Patentrezept geben, einige häufig in C64-Programmen vorkommende POKE-Adressen sollen aber im folgenden Teil etwas ausführlicher besprochen werden.

POKE-Befehle für Farbe und Bildschirm

Selbst die einfachsten Programme enthalten in der Regel Befehle, um die Rahmen- und Hintergrundfarbe des Bildschirms einzustellen. Während es beim C16 zu diesem Zweck den COLOR-Befehl gibt, arbeiten C64 und VC 20 hier mit POKE-Befehlen. Beim VC 20 werden beide Einstellungen (Rahmenfarbe und Hintergrundfarbe) gleichzeitig mit einem einzigen POKE-Befehl in Register 36879 durchgeführt (Tabelle 2). Der C64 verwendet dafür zwei getrennte Register, nämlich 53280 für die Rahmenfarbe und 53281 für die Hintergrundfarbe. Beim C16 schließlich geht es etwas komfortabler zu. Tabelle 3 zeigt die Anwendung des COLOR-Befehls bei diesem Computer. Bei allen drei Geräten sind jeweils 16 Farben möglich (Tabelle 4). Zum Beispiel erzeugen die Befehle »POKE 53280,7:POKE 53281,5« beim C64 einen gelben Bildschirmrahmen und einen grünen Hintergrund. Mit »POKE 36879,95« erreicht man dasselbe beim VC 20, mit »COLOR 4, 8 : COLOR 0,6« beim C16. Die Farbnummern sind beim C16 immer um Eins erhöht gegenüber den entsprechenden Nummern beim C64.

Der Bildschirmspeicher des C64 belegt die Adressen 1024 bis 2023 (1000 Zeichen), der zugehörige Farbspeicher geht von 55296 bis 56295. Mit »POKE 1024, 1: POKE 55296,2« erscheint zum Beispiel beim C64 ein rotes »A« in der linken oberen Bildschirmecke. Beim C16 ist der Video-Speicher etwas höher angesiedelt. Er belegt die Adressen 3072 bis 4071, das Farb-RAM liegt von Adresse 2048 bis 3047. Der für den Anwender verfügbare Basic-Speicherbereich beginnt beim C64 bei Adresse 2048, beim C16 bei Adresse 4096. Tabelle 5 zeigt die Speicheraufteilung von C64, C16 und VC 20.

Beim VC 20 liegen die Verhältnisse noch etwas komplizierter: Die Anfangsadressen von Bildschirm- und Farbspeicher sind nämlich je nach Speicherausbau unterschiedlich. In der Grundversion und mit der 3-KByte Erweiterung beginnt das Video-RAM bei Adresse 7680 und geht bis zur Adresse 8185. Das Farb-RAM belegt dann den Bereich von 38400 bis 38905. In dieser Konfiguration liegt der Bildschirmspeicher oberhalb des für den Anwender verfügbaren RAM-Bereichs, der ab Adresse 4096 (Grundversion) beziehungsweise 1024 (3-KByte-Erweiterung) beginnt. Sobald jedoch beim VC 20 eine Speichererweiterung von mindestens 8 KByte eingesteckt ist, wandert das Video-RAM nach unten und beginnt dann bei Adresse 4096. Dies geschieht, um für Basic-Programme einen zusammenhängenden Speicherbereich von der Adresse 4608 an aufwärts zu schaffen. Eine eventuell vorhandene zusätzliche 3-KByte-Erweiterung kann in diesem Fall nicht für Basic-Programme genutzt werden. Natürlich verändert auch das Farb-RAM noch seine Lage und startet jetzt bei Adresse 37888.

Die musikalische Seite

Einige beim C64 ebenfalls häufig vorkommende POKE-Adressen wurden bisher noch gar nicht erwähnt. Gemeint sind die zur Programmierung von Musik und Geräuscheffekten benutzten Register. Leider ist die Art der Tonerzeugung beim C64 auf der einen und VC 20/C16 auf der anderen Seite völlig unterschiedlich, so daß sich keine äquivalenten Befehle angeben lassen. Wie bereits zu Anfang erwähnt, verfügt der C64 über einen vollwertigen, dreistimmigen Synthesizer-Baustein, während der C16 und VC 20 sich mit einfachen Tongeneratoren zufrieden geben müssen. Tabelle 6 enthält zur Referenz eine Übersicht über die beim C64 zur Tonerzeugung benutzten Register. Diese Tabelle dient allerdings wirklich nur zur Orientierung, denn eine Simulation des C64-Synthesizers ist weder mit dem VC 20 noch mit dem C16 möglich. Es empfiehlt sich daher, bei der Programmanpassung zunächst einmal alle derartigen POKE-Befehle fortzulassen und später eigene Sound-Routinen einzufügen. C64-Besitzer haben es umgekehrt natürlich etwas einfacher. Mit etwas Geschick und einem guten Handbuch zur Musik-Programmierung können Sie dem C64 natürlich auch die VC20/C16-Töne entlocken. Für alle C64- und C16-Besitzer, die vielleicht doch einmal in die Verlegenheit kommen, das eine oder andere VC 20-Programm auf ihre Geräte umzuschreiben, sind in Tabelle 7 die Notenwerte und die Tongenerator-Adressen des VC 20 angegeben.

Joysticks und Paddles

Der C64 verfügt ebenso wie der C16 über zwei Joystick-Ports, während dem VC 20 nur einer vergönnt ist. Die Abfrage ist beim C16 wieder am komfortabelsten, nämlich per Basic-Funktion. »PRINT JOY(1)« ergibt den Richtungswert von Joystick 1, entsprechend wird mit JOY(2) die Richtung des zweiten Joysticks abgefragt. Beim C64 entsprechen diesen Abfragen PEEKs in die Speicherstellen 56320 (Joystick 1) und 56321 (Joystick 2). Ziemlich merkwürdig gelöst ist die Joystick-Abfrage beim VC 20, wo ein recht kompliziertes Abfrageprogramm erforderlich ist. Der Feuerknopf und die Schalter 0, 1 und 2 des Joysticks werden nämlich beim VC 20 über die VIA 1 gelesen, während der Zustand von

Schalter 3 über VIA 2 abgefragt wird. Normalerweise sind Joystick-Abfragen in Programmen leicht zu finden. Halten Sie beim VC 20 nach PEEKs in die Speicherstellen 37137 und 37152 Ausschau und beim C64 nach entsprechenden Abfragen der Adressen 56320 und 56321. Während die Joystick-Abfrage des C64 noch relativ leicht an den C16 angepaßt werden kann (»PEEK(56320)« entspricht »JO>(1)«, wird es beim VC 20 schon schwieriger. In der Regel wird man die Joystick-Abfrage völlig neu schreiben müssen, da die Art der Abfragen beim VC 20 einerseits und C16/C64 andererseits zu unterschiedlich ist. Bei der Steuerung mittels Drehreglern (sogenannten Paddles) ist die Anpassung wenigstens im Falle des VC 20 wesentlich einfacher zu realisieren. Die Paddle-Werte werden beim C64 aus den Registern 54297 und 54298 mit PEEK ausgelesen.

Wo sich das Umschreiben lohnt

Beim VC 20 sind es die Register 36872 und 36873, beim C16 ist eine Paddle-Abfrage nicht vorgesehen und muß, falls überhaupt sinnvoll, per Joystick-Abfrage simuliert werden.

An dieser Stelle muß noch einmal deutlich darauf hingewiesen werden, daß die Zahl der C64-Programme, die sich mit vertretbarem Aufwand zum VC 20 oder C16 übertragen lassen, doch verhältnismäßig gering ist. VC20-Besitzer sind hier etwas besser dran. Sie finden in diesem Sonderheft ein Programm, das Maschinenprogramme für den C64 in entsprechende Routinen für den VC20 übersetzt. Es funktioniert in etwa 70-80% aller Fälle. Im allgemeinen ist jetzt speziell bei Spielprogrammen in der Regel äußerste Vorsicht geboten, da hier aus Geschwindigkeitsgründen zumeist mit verschiedenen Routinen in Maschinensprache gearbeitet wird. Von Sprites und hochauflösender Grafik einmal ganz zu schweigen. Dagegen gibt es viele Anwendungsprogramme, die den C64 nicht als hochspezialisierte Spielmaschine (die er ohne Zweifel ist) ausnutzen, sondern ganz einfach Problemlösungen in Basic anbieten. Dabei kann es sich beispielsweise um eine Dateiverwaltung, einen Vokabeltrainer oder ganz einfach ein Programm zum Ausdrucken eines Jahreskalenders handeln. Für fast alle derartigen Programme sollte es möglich sein, eine Anpassung mit Hilfe der hier abgedruckten Tabellen und Hinweise vorzunehmen. (ev)

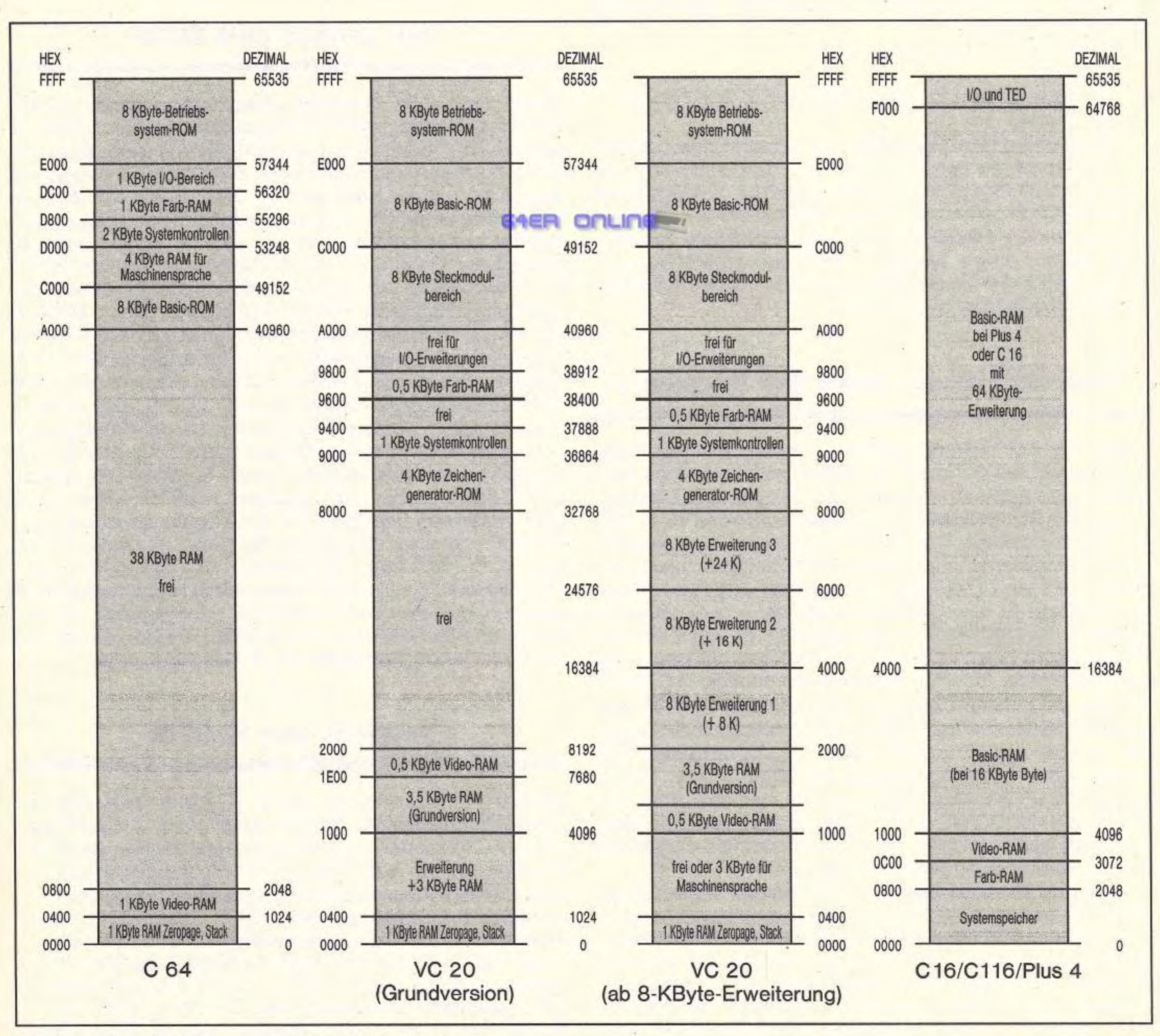


Tabelle 5. Speicherbelegung bei C64, C16 und VC 20

	C64	VC 20 Grundversion	VC 20 ab 8 KByte	C16/C116/Plus 4
Video-RAM	1024	7680	4096	3072
freier RAM-Bereich	2048	4096	4608	4096
Video-Chip (VIC/TED)	53248	36864	36864	65280
Synthesizer-Chip	54272	–	–	–
Farb-RAM	55296	38400	37888	2048
Basic-ROM	40960	49152	49152	32768 (in ROM-Bank)

Tabelle 1. Wichtige Speicheradressen bei C64, C16 und VC 20 zur ersten Orientierung

COLOR-Farbzonen-Nr., Farb-Nr., [, Helligkeitswert]

Mit der COLOR-Anweisung wird eine der fünf Farbzonen bestimmt:

Farbzonen-Nr.	Bezeichnung
0	Bildschirm-Hintergrund
1	Vordergrund (Buchstaben, Zeichen)
2	Mehrfarben 1
3	Mehrfarben 2
4	Bildschirmrand

Die zweite Zahl in der COLOR-Anweisung selektiert die Hintergrundfarbe des gewählten Bildschirmbereichs. Diese Zahl ist mit den Farbtasten der Tastatur identisch.

Farb-Nr.	Farbe	Farb-Nr.	Farbe
1	Schwarz	9	Orange
2	Weiß	10	Braun
3	Rot	11	Gelb/Grün
4	Cyan	12	Rosa
5	Purpur	13	Blau/Grün
6	Grün	14	Hellblau
7	Blau	15	Dunkelblau
8	Gelb	16	Hellgrün

Jede Farbe ist auch noch in ihrer Helligkeit (Luminanz) veränderbar. Im Anhang an die Farb-Nr. kann diese Luminanz von Null (0= dunkel) bis sieben (7= hell) variieren. Der Standardwert für die Helligkeit ist 7. Die acht Helligkeitsstufen gelten für alle Farben, mit Ausnahme von Schwarz.

Tabelle 3. Der COLOR-Befehl des C16 kann bis zu drei Parameter enthalten (vergleiche Tabelle 4)

VC 20 und C64		nur C64	
0 schwarz	4 violett	8 orange	12 grau 2
1 weiß	5 grün	9 braun	13 hellgrün
2 rot	6 blau	10 hellrot	14 hellblau
3 türkis	7 gelb	11 grau 1	15 grau 3

Tabelle 4. Codierung einzelner Farben bei C64 und VC 20

Adresse Stimme 1	Stimme 2	Stimme 3	Inhalt
54272	54279	54286	Tonfrequenz (Low-Byte)
54273	54280	54287	Tonfrequenz (High-Byte)
54274	54281	54288	Tastaturverhältnis Rechteckgenerator (Low)
54275	54282	54289	Tastaturverhältnis Rechteckgenerator (High)
54276	54283	54290	Wellenform
54277	54284	54291	Anschlag/Abschwellen
54278	54285	54292	Halten/Ausklingen
54296			Lautstärke
Mögliche Wellenformen: Dreieck (17), Sägezahn (33), Rechteck (65), Rauschen (129).			

Tabelle 6. Die wichtigsten Register des C64-Soundchips

Rahmen ▶	BLK	WHT	RED	CYAN	PUR	GRN	BLU	YEL
Hintergrund ▼	SCHW	WEISS	ROT	TÜRKIS	VIOLETT	GRÜN	BLAU	GELB
SCHWARZ	8	9	10	11	12	13	14	15
WEISS	24	25	26	27	28	29	30	31
ROT	40	41	42	43	44	45	46	47
TÜRKIS	56	57	58	59	60	61	62	63
VIOLETT	72	73	74	75	76	77	78	79
GRÜN	88	89	90	91	92	93	94	95
BLAU	104	105	106	107	108	109	110	111
GELB	120	121	122	123	124	125	126	127
ORANGE	136	137	138	139	140	141	142	143
HELLORANGE	152	153	154	155	156	157	158	159
ROSA	168	169	170	171	172	173	174	175
HELLTÜRKIS	184	185	186	187	188	189	190	191
HELLVIOLETT	200	201	202	203	204	205	206	207
HELLGRÜN	216	217	218	219	220	221	222	223
HELLBLAU	232	233	234	235	236	237	238	239
HELLGELB	248	249	250	251	252	253	254	255

Tabelle 2. Farbkombinationen für Bildschirmrahmen und Hintergrund beim VC 20. Die einzelnen Zahlenwerte errechnen sich nach der Formel »POKE-Wert = Hintergrund x 16 + Rahmen + 8« (vergleiche Tabelle 4).

NOTE	WERT	NOTE	WERT
C	135	G	215
C#	143	A^b	217
D	147	A	219
E^b	151	B	221
E	159	H	223
F	163	C	225
F#	167	C#	227
G	175	D	228
A^b	179	E^b	229
A	183	E	231
B	187	F	232
H	191	F#	233
C	195	G	235
C#	199	A^b	236
D	201	A	237
E^b	203	B	238
E	207	H	239
F	209	C	240
F#	212	C#	241

Stimmlagen-Befehle	X=	Funktion
POKE 36878,X	0 bis 15	setzt die Lautstärke
POKE 36874,X	128 bis 255	spielt Note
POKE 36875,X	128 bis 255	spielt Note
POKE 36876,X	128 bis 255	spielt Note
POKE 36877,X	128 bis 255	Geräuscheffekte

Tabelle 7. Notenwerte und Tongenerator-Adressen beim VC 20. Diese Tabelle dient zur Orientierung für C 16-Besitzer, die das eine oder andere VC 20-Programm für ihren Computer umsetzen möchten.

Die Datasette streikt nie wieder

Einer der häufigsten Fehler, der bei der Datasette auftritt, ist ein verstellter Tonkopf. Dieser Fehler macht sich besonders dann bemerkbar, wenn mit Turbo Tape oder ähnlichen Programmen gearbeitet wird. Mit der hier beschriebenen Schaltung läßt sich extrem einfach, ohne jegliches Programm, der Tonkopf an jede Datenkassette anpassen.

Um die Datasette oder einen anderen Datenrecorder zu justieren, gibt es grundsätzlich zwei Möglichkeiten. Eine kleine elektronische Schaltung, mit der sich unabhängig vom Computer die Tonkopfstellung an jede Datenkassette anpassen läßt und ein Programm, das in irgendeiner Form die Tonkopfstellung grafisch auf dem Monitor des Computers darstellt. Ein solches Programm ist aber unbrauchbar, egal wie gut oder schlecht es ist. Der Grund dafür ist ganz einfach der, daß sich nach erfolgter Justage Programme, die zuvor auf anderen Kassetten gespeichert wurden, nicht mehr laden lassen; unter anderem auch das Justageprogramm selbst. Sollen solche Programme geladen werden, müßte das Justageprogramm noch einmal abgetippt werden.

Um das zu vermeiden, stellen wir Ihnen eine Schaltung vor, mit der das Einstellen extrem einfach wird.

Damit die Schaltung verständlich wird, zuerst ein paar Worte zur Datasetten-Elektronik.

Theorie und Praxis

Sie besteht aus zwei Hauptgruppen, einem zweistufigen Verstärker, der die Aufgabe hat, das analoge Signal, das vom Tonkopf kommt, zu verstärken.

Analog deshalb, weil sich digitale Signale nicht auf Band speichern lassen. Selbst wenn ein solches Signal am Tonkopf anliegt, wird es nicht als solches auf das Band geschrieben, sondern in Form einer Sinusschwingung. Beim Laden muß diese Sinusschwingung wieder in eine Form gebracht werden, die der Computer versteht. Folglich muß die Sinusschwingung in ein Rechtecksignal gewandelt werden.

Dies geschieht in der zweiten Hauptstufe mit Hilfe eines Schmitt-Triggers. Am Ausgang des Schmitt-Triggers liegt das Signal in Form einer Rechteckschwingung vor, die entweder einen Spannungspegel von 0 oder 5 Volt hat. Dieses Signal eignet sich nicht zur Einstellung des Tonkopfes, weil die Amplitude des Signals, unabhängig von der Tonkopfstellung, immer konstant zwischen 0 und 5 Volt hin- und herspringt.

Die Messung mit einem Oszilloskop ergab aber, daß, abhängig von der Tonkopfstellung, die Amplitude der analogen Spannung schwanke.

Ist der Tonkopf optimal eingestellt, geht die Amplitude der Spannung gegen ein Maximum. Ist der Tonkopf dejustiert, weicht die Amplitude, abhängig von der Tonkopfstellung, vom Maximum ab. Man kann es jedoch keinem Datasetten-Besitzer zumuten, sich ein Oszilloskop anzuschaffen, nur um die Datasette zu justieren.

Die vorliegende Bastelanleitung, deren Bauteile zu einem Preis von unter fünf Mark zu haben sind, ersetzt in diesem Fall ein Oszilloskop. Mit der Schaltung (Bild 1) läßt sich eine Spannung, natürlich in gewissen Grenzen, auf Maximum abgleichen. Das Herz ist ein Operationsverstärker vom Typ LF 356, der als Komperator (Schwellwertschalter) betrieben wird.

Bastelanleitung

Außerdem hat dieser Operationsverstärker gegenüber anderen den Vorteil, daß seine Eingangsstufe aus einem Feldeffekttransistor besteht. Der Eingangswiderstand geht dadurch gegen unendlich und belastet das zu messende Signal in keinster Weise. Mit dem Trimmpotentiometer läßt sich eine Schwellspannung (Bild 2) einstellen, die laufend mit der analogen Sinusschwingung verglichen wird.

Ist der Momentanwert der Sinusschwingung kleiner als die vorgegebene Schwellspannung, führt der Ausgang des LF

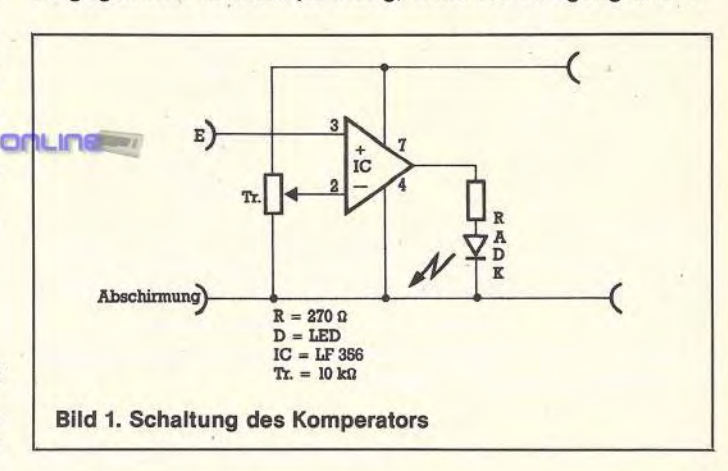


Bild 1. Schaltung des Komperators

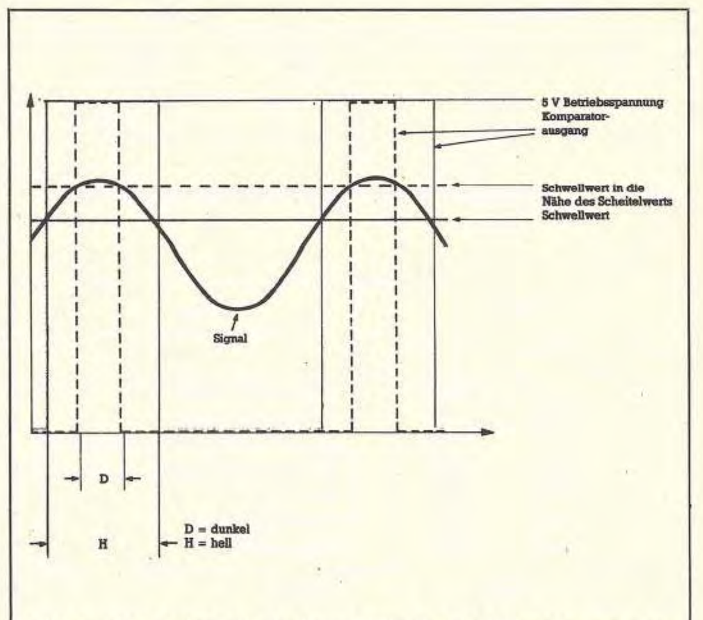


Bild 2. Die Helligkeit der Leuchtdiode ist abhängig von der Zeitspanne, die der Komperatorausgang auf +5V liegt.

356 0 Volt. Wird der Momentanwert größer, springt der Ausgang des LF 356 auf +5 Volt und regt dadurch eine Leuchtdiode an. Wird die Schwellspannung in den Scheitelpunkt der Sinusschwingung gelegt (gestrichelte Linie in Bild 2), geht die Zeitspanne, in der der Ausgang des Komperators auf 5 Volt liegt, gegen ein Minimum. Daraus folgt, daß die Helligkeit der Leuchtdiode abnimmt, je näher die Schwellspannung an den Scheitelwert der Sinusschwingung rückt. Wird dagegen die Amplitude des Signals, also der Sinusschwingung, vergrößert, wird die Helligkeit der Leuchtdiode wieder größer. Denn die Zeitspanne, in der der Ausgang des Komperators auf 5Volt liegt, wird größer. Dieses ist vom Prinzip her der ganze Abgleichvorgang. Mit dem Trimmpotentiometer wird auf minimale Helligkeit und mit der Tonkopfeinstellschraube auf maximale Helligkeit abgeglichen.

Aufgebaut wird die Schaltung auf einer kleinen Lochrasterplatine. Diejenigen, die sich eine Platine ätzen wollen, finden das Layout im Verhältnis 1:1 in Bild 3. Wie die einzelnen Pins der Bauelemente miteinander verbunden werden, zeigt Bild 4. Achten Sie beim Zusammenbau auf die richtige Polarität der Leuchtdiode (Bild 5).

Ist die Schaltung zusammengelötet, muß sie noch im Datasettengehäuse untergebracht werden. Öffnen Sie dazu die Datasette und bohren an einer geeigneten Stelle ein Loch in das Gehäuseoberteil, so daß die Leuchtdiode gerade in dieses Loch paßt.

Verbinden Sie die Anschlüsse »+« und »—« (Bild 4) mit den Motoranschlußklemmen. Dabei ist ebenfalls auf die Polarität zu achten. Im allgemeinen ist sie auf dem Motor gekennzeichnet. Der in Bild 4 gekennzeichnete Punkt »E« (für Eingang) muß über ein abgeschirmtes Kabel mit einem der beiden Lötpunkte auf der Datasettenplatine (Bild 6) verbunden werden. Die Abschirmung ist an den mit »—« gekennzeichneten Punkt (Bild 4) zu löten.

Bei den beiden Lötpunkten handelt es sich um den Ausgang des ersten beziehungsweise zweiten Analogverstärkers einer Commodore-Datasette.

Geräte anderer Hersteller sind zum Teil anders aufgebaut. Es kann vorkommen, daß die beiden in Bild 6 gekennzeichneten Analogverstärker in einem Gehäuse untergebracht sind. In diesem Fall ist der Punkt »E« mit dem Pin 8 dieses ICs zu verbinden.

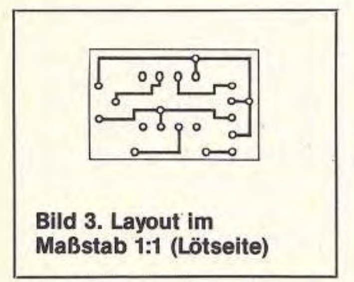


Bild 3. Layout im Maßstab 1:1 (Lötseite)

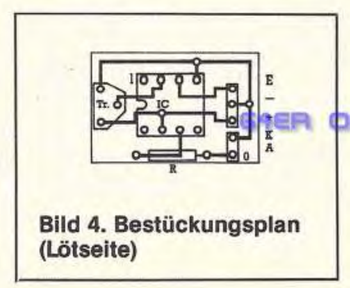


Bild 4. Bestückungsplan (Lötseite)

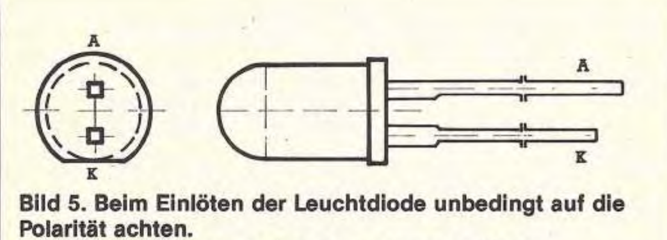


Bild 5. Beim Einlöten der Leuchtdiode unbedingt auf die Polarität achten.
Anode=A=längeres Beinchen oder runde Seite.

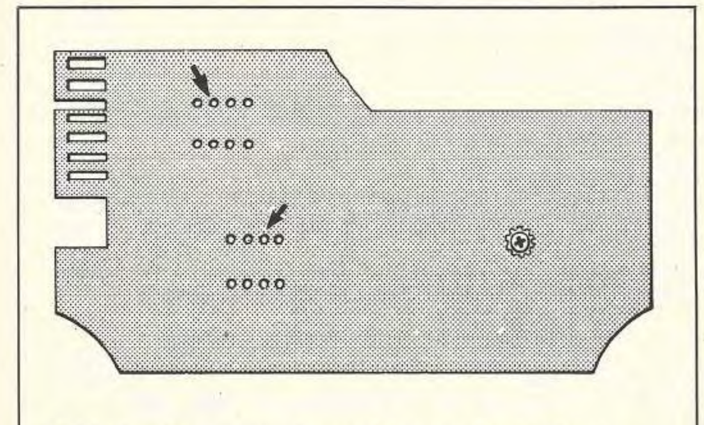


Bild 6. An einem der gekennzeichneten Punkte ist der Punkt »E« (Bild 4) zu löten.

Einfaches Einstellen

Bevor die Datasette zusammengebaut wird, ist die Schaltung an die Datasetten-Elektronik anzupassen. Schalten Sie dazu den C 64 ein, legen eine Programm-Kassette in die Datasette und drücken die PLAY-Taste.

Nach der direkten Eingabe POKE 54272+24,15 drehen Sie den Lautstärkeregler Ihres Monitors auf Maximum und warten, bis die Übertragung des Programm- oder Datencodes im Lautsprecher zu hören ist. Nun muß in einem wechselseitigen Einstellvorgang die Helligkeit der Leuchtdiode am Trimmpotentiometer auf Minimum und an der Tonkopfeinstellschraube auf Maximum abgeglichen werden. Bei Commodore-Datasetten befindet sich die Tonkopfeinstellschraube (Kreuzschlitz) bei gedrückter PLAY-Taste unter einem etwa 5 mm großen Loch auf dem Gehäuseoberteil. Soll eine andere Datasette justiert werden, muß der Kassettendeckel abgebaut werden. Die Tonkopfschraube ist nun eine der beiden Tonkopfbefestigungsschrauben und zwar die, an der sich eine Spiralfeder befindet. Doch nun zum Abgleichvorgang. Dazu gehen Sie bitte folgendermaßen vor:

1. Am Trimmpotentiometer drehen, bis die Leuchtdiode schwach flackert. Dadurch wird die Schwell- oder Schaltspannung in den Scheitelpunkt der Sinusschwingung gelegt.
2. An der Tonkopfeinstellschraube drehen, bis die Helligkeit der Leuchtdiode ein Maximum erreicht hat. Dadurch wird die Amplitude des Signals, das vom Tonkopf kommt, auf Maximum abgeglichen.

Noch ein Tip

Der letzte Punkt ist nur dann erforderlich, wenn die Datasette nicht optimal eingestellt war, beziehungsweise eine Kassette benutzt wird, die mit einer anderen Datasette beschrieben wurde. In diesem Fall muß der Einstellvorgang solange wiederholt werden, bis eine Einstellung erreicht ist, bei der die Leuchtdiode erlischt, sobald der Tonkopf minimal verstellt wird. Bauen Sie nun die Datasette wieder zusammen. Schalten Sie voher aber den C 64 aus.

Wollen Sie jetzt ein Programm laden, das mit einem dejustierten Tonkopf aufgenommen wurde, brauchen Sie nur noch, nachdem der C 64 eingeschaltet wurde, die Kassette einzulegen, die PLAY-Taste zu drücken und so lange an der Tonkopfeinstellschraube zu drehen, bis die Helligkeit der Leuchtdiode ein Maximum erreicht hat.

Zum Schluß soll noch darauf hingewiesen werden, daß selbst bei Commodore-Datasetten die unterschiedlichsten Platinen existieren. Befinden sich auf Ihrer Platine nur zwei 14beinige IC's, dann ist der Punkt »E« an den Pin 8 oder 13 jenes IC's zu löten, das sich auf der linken Platinenseite befindet (vorausgesetzt, Sie haben die Platine so vor sich liegen, wie Bild 6 zeigt). (ah)

C 64-Programme für den VC 20

Mit diesem Programm ist es möglich, auch C64-Programme auf Ihrem VC20 laufen zu lassen.

Das Programm »Reloctable« konvertiert Maschinenspracheroutinen, die für den C64 geschrieben worden sind, spielend leicht in ablauffähige Programme auf dem VC20. Es werden pro Sekunde etwa 50 Byte übersetzt und in den Speicher des VC20 geschrieben.

Zuerst muß die zu übersetzende Maschinenroutine in den Speicher Ihres VC20 gebracht werden. Dabei darf nur das von Basic aus erreichbare RAM beschrieben werden. Es ist dabei völlig unerheblich, an welche Stelle das Programm geschrieben (gePOKEt) wird, denn meist sind die Adreßräume des C64 für den VC20 sowieso nicht erreichbar. Aber bedenken Sie, daß das Übersetzungsprogramm noch nachgeladen werden muß (am besten den RAM-Top heruntersetzen).

Jetzt wird das Übersetzungs-Programm nachgeladen und mit RUN gestartet. Kurze Zeit später wird nach der Länge des zu bearbeitenden Programmes gefragt. Nach Eingabe der Bytezahl wird die Adresse (dezimal) erfragt, ab der das Maschinenprogramm steht. Eine weitere Abfrage klärt, ob, und wenn ja, wohin das Programm ablauffähig verschoben werden soll. Gibt man an dieser Stelle einfach RETURN ein, wird auf den Übersetzungsmodus von C64 auf VC20 umgeschaltet. Schließlich muß noch die Anfangsadresse eingegeben werden, ab der das Programm eigentlich im C64-Speicher stehen sollte. Mit der Meldung »Übersetzungsversuch« beginnt die Routine zu arbeiten. Nach READY kann das Programm mit `SYS Anfangsadresse` gestartet werden (Einsprungpunkt beim C64 beachten!).

Die Übersetzung klappt in 90 Prozent aller Fälle. Bei den restlichen 10 Prozent handelt es sich um Maschinenprogramme, die den SID oder die Grafik des C64 direkt ansprechen. Da vergleichbare Adressen beim VC20 nicht vorhanden sind, mußte an dieser Stelle ein Kompromiß eingegangen werden. Das Umsetzungsprogramm reagiert in diesem Fall mit einer Meldung, daß die Übersetzung kritisch wird. Trotzdem kann man die Übersetzung starten, ohne einen Computerabsturz nach dem Start des behandelten Programmes befürchten zu müssen. An den entsprechenden, nicht konvertierbaren Stellen werden Unterprogramme eingesetzt. Ein Versuch lohnt sich allerdings immer. So gelingt die Übersetzung von Basic-Erweiterungen fast ausnahmslos. (Der Autor übersetzte nahezu die gesamte Maschinenspracheroutinesammlung aus »C64 Tips und Tricks«).

Eine gleichwohl interessante Möglichkeit des Reloctable-Programms besteht im ablauffähigen Verschieben von VC20-Maschinenroutinen. Mit diesem Programmteil wird es leicht möglich, Speicherbereiche inklusive Adreßumrechnung frei im RAM zu verschieben. Sicher eine willkommene Hilfe bei der Zusammenstellung von Maschinensprache-Programmpaketen ohne die sonst üblichen lästigen ungenutzten Bereiche. Die Verschiebung klappt in mindestens 95 Prozent aller Fälle. Lediglich Maschinenprogramme, innerhalb derer Listen oder Strings abgelegt sind, erfordern eine gewisse »Nachbehandlung« mit einem Disassembler und dem manuellen Setzen der relevanten Bytes. Das läßt sich aber mit etwas Übung ganz gut machen. Aber auch ohne irgendwelche Maschinenkenntnisse ist die Trefferwahrscheinlichkeit des Programms groß genug. Übrigens können auch die übersetzten C64-Programme im zweiten Schritt frei verschoben werden. Für eine sinnvolle Arbeit mit dem vorliegenden Programm ist mindestens eine 8 KByte-Erweiterung notwendig.

(Manfred Zimmermann/dm)

```
100 REM REOCTABLE                          <139>
110 REM TRANSLATION                        <253>
120 REM C. BY M.ZIMMERMANN,WESEL           <115>
130 REM                                    <192>
140 REM                                    <202>
150 PRINT"{CLR,2DOWN}**********************
    *{DOWN}                                <246>
160 PRINT"{5SPACE}RELOCTABLE{2DOWN}"       <250>
170 PRINT"************************"        <167>
180 DIM OP%(255)                           <087>
190 FOR T=0 TO 255                         <120>
200 READ A%                                <066>
210 OP%(T)=A%                              <089>
220 NEXT                                   <230>
230 REM                                    <036>
240 REM STANDORTE                          <088>
250 REM                                    <056>
260 INPUT"{DOWN}ANZ. D.. BYTES ";AN        <206>
270 IF AN=0 THEN 260                       <191>
280 INPUT"{DOWN}STARTADRESSE ";SA          <255>
290 ZI=SA                                  <188>
300 INPUT"{DOWN}ZIELADRESSE ";ZI           <149>
310 IF ZI<>SA THEN 430                     <083>
320 TR$="J"                                <140>
330 INPUT"{DOWN}START C-64 ";TS            <027>
340 DIM C%(211),V%(211)                    <232>
350 FOR T=0 TO 210 STEP 2                  <067>
360 READ CH,CL,VH,VL                       <233>
370 C%(T)=CH:C%(T+1)=CL                    <026>
380 V%(T)=VH:V%(T+1)=VL                    <179>
390 NEXT                                   <146>
400 BS=4*(PEEK(36866)AND 128)+64*(PEEK(368
    69)AND 120)                            <245>
410 FS=37888+4*(PEEK(36866)AND 128)        <208>
420 PRINT:PRINT CHR$(18)"UEBERSETZUNGSVERS
    UCH!!"CHR$(146):GOTO 510               <200>
430 IF ZI>827 AND ZI+AN<1021 THEN 480      <048>
440 IF ZI<PEEK(49)+256*PEEK(50)OR ZI+AN+6>
    49151 THEN PRINT"{DOWN}UNGUELTIGES ZIE
    L{2SPACE,2DOWN}":GOTO 300              <194>
450 IF SA<=ZI THEN IF SA+AN+1>ZI THEN PRIN
    T"UNGUELTIGES ZIEL{2DOWN}":GOTO 300    <181>
460 IF ZI<=SA THEN IF ZI+AN+1>SA THEN PRIN
    T"UNGUELTIGES ZIEL{2DOWN}":GOTO 300    <235>
470 PRINT:PRINT CHR$(18)"VERSCHIEBUNGSVERS
    UCH!!"CHR$(146)                        <083>
480 FOR T=0 TO AN-1                        <237>
490 POKE ZI+T,PEEK(SA+T)                   <023>
500 NEXT                                   <000>
510 REM                                    <062>
520 REM ANALYSE                            <252>
530 REM                                    <084>
540 PRINT:PRINT"WARTEN AUF "CHR$(18)"READY
    "CHR$(146)" !!"                        <199>
550 FOR T=0 TO AN-1                        <053>
560 WE=PEEK(SA+T)                          <033>
570 IF KK$<>"J"THEN 640                    <050>
580 IF WE=201 OR WE=224 OR WE=192 THEN 600 <165>
590 GOTO 660                               <146>
600 H=PEEK(ZI+T+1)                         <148>
610 IF H>26 AND H<42 THEN H=22+H-40:POKE Z
    I+T+1,H:GOTO 660                       <170>
620 IF H>18 AND H<27 THEN H=23+H-25:POKE Z
    I+T+1,H                                <221>
630 GOTO 660                               <186>
640 IF WE=169 OR WE=162 OR WE=160 THEN GOS
```

Listing. »Reloctable«

```
    UB 970                                       <177>
650 IF OP%(WE)=2 THEN GOSUB 880                  <011>
660 T=T+OP%(WE)                                  <161>
670 NEXT                                         <172>
680 IF K$<>"J"THEN0800                           <095>
690 IF K$="J"THEN K$="":KK$="J"                  <079>
700 PRINT"{CLR,2DOWN}DAS PROGRAMM VER-{DOW
    N}"                                          <075>
710 PRINT"WENDETE ADRESSEN DES{DOWN}"            <242>
720 PRINT"BILDSCHIRM- UND FARB-{DOWN}"           <047>
730 PRINT"BEREICHS.C64 UND VC20{DOWN}"           <182>
740 PRINT"UNTERSCHEIDEN SICH DA{DOWN}"           <228>
750 PRINT"WESENTLICH.EIN UEBER-{DOWN}"           <134>
760 PRINT"SETZUNGSVERSUCH IST{DOWN}"             <222>
770 PRINT"KRITISCH ! SOLL ICH ES"                <031>
780 PRINT"VERSUCHEN?{2DOWN}":INPUT"(J/N) "
    ;A$                                          <175>
790 IF A$="J"THEN 550                            <138>
800 DH=INT(ZI/256)                               <003>
810 DL=ZI-256*DH                                 <102>
820 POKE ZI+AN+2,DL                              <110>
830 POKE ZI+AN+3,DH                              <058>
840 POKE ZI+AN+4,197                             <162>
850 POKE ZI+AN+5,206                             <006>
860 POKE ZI+AN+6,196                             <154>
870 END                                          <110>
880 REM                                          <180>
890 REM 3 BYTE-BEFEHL                            <197>
900 REM                                          <200>
910 H=PEEK(SA+T+1)+256*PEEK(SA+T+2)              <063>
920 G=T                                          <149>
930 IF TR$="J"THEN GOSUB 1240                    <244>
940 IF H<SA OR H>SA+AN-1 THEN RETURN             <101>
950 GOSUB 1510                                   <000>
960 RETURN                                       <000>
970 REM                                          <014>
980 REM LDA /LDX/ LDY                            <088>
990 REM                                          <034>
1000 IF T=G AND T<>0 THEN RETURN                 <225>
1010 G=T:GE=WE:KK=0                              <208>
1020 H1=PEEK(SA+T+1)                             <086>
1030 G=G+OP%(GE)+1                               <115>
1040 IF G>AN OR KK>1 THEN G=T:RETURN             <109>
1050 GE=PEEK(SA+G)                               <155>
1060 IF GE=169 OR GE=162 OR GE=160 THEN 10
     80                                          <015>
1070 KK=KK+1:GOTO 1030                           <065>
1080 H2=PEEK(SA+G+1)                             <104>
1090 H=H1+256*H2                                 <037>
1100 IF TR$="J"THEN GOSUB 1240                   <160>
1110 IF H<SA OR H>SA+AN-1 THEN RETURN            <017>
1120 GOSUB 1510                                  <172>
1130 RETURN                                      <172>
1140 REM                                         <186>
1150 REM BETRIEB                                 <181>
1160 REM                                         <206>
1170 L=0:H1=0:H2=0:IF H=58592 OR H=58602 T
     HEN 1610                                    <222>
1180 H2=256*C%(L)+C%(L+1)                        <157>
1190 IF H2=H THEN H=256*V%(L)+V%(L+1):RETU
     RN                                          <020>
1200 IF H2>H THEN H=256*V%(L-2)+V%(L-1)+H-
     H1:RETURN                                   <123>
1210 H1=H2:L=L+2                                 <163>
1220 IF L<211 THEN 1180                          <082>
1230 RETURN                                      <016>
1240 REM                                         <030>
1250 REM UEBERSETZUNG                            <215>
1260 REM                                         <050>
1270 N$=""                                       <079>
1280 REM ZERO                                    <139>
1290 IF H=784 OR H=785 OR H=786 THEN H=H-7
     84:GOTO 1510                                <226>
1300 REM BILDSCHIRM                              <163>
1310 IF H>1023 AND H<2024 THEN H=BS+H-1024
     :K$="J":GOTO 1510                           <235>
1320 REM SPRITES                                 <192>
1330 IF H>2039 AND H<2048 THEN N$="J"            <181>
1340 REM INTERPRET 1                             <116>
1350 IF H>40959 AND H<49151 THEN H=H+8192:
     GOSUB 1590:GOTO 1510                        <195>
1360 REM MASCH. RAM                              <181>
1370 IF H>TS-1 AND H<TS+AN THEN H=SA+H-TS:
     GOTO 1510                                   <043>
1380 IF H>49151 AND H<53248 THEN N$="J"          <221>
1390 REM VIDEO/SOUND                             <164>
1400 IF H>53247 AND H<55295 THEN N$="J"          <155>
1410 REM INTERP. 2                               <112>
1420 IF H>55295 AND H<56296 THEN H=FS+H-55
     296:K$="J":GOTO 1510                        <144>
1430 REM BETR. UNKLAR                            <087>
1440 IF H>56295 AND H<57344 THEN N$="J"          <126>
1450 REM INTERPR. 2                              <097>
1460 IF H>57343 AND H<58463 THEN H=H-3:GOS
     UB 1590:GOTO 1510                           <184>
1470 IF H>58462 AND H<65418 THEN GOSUB 117
     0:GOTO 1510                                 <191>
1480 REM NOTLOESUNG                              <224>
1490 IF N$="J"THEN DL=234:DH=DL:POKE ZI+T,
     DH:POKE ZI+G,DH:GOTO 1540                   <065>
1500 RETURN                                      <032>
1510 DD=H+ZI-SA                                  <037>
1520 DH=INT(DD/256)                              <242>
1530 DL=DD-256*DH                                <054>
1540 IF G=T THEN G=T+1                           <070>
1550 POKE ZI+T+1,DL                              <153>
1560 POKE ZI+G+1,DH                              <063>
1570 IF G=T+1 THEN G=0                           <013>
1580 RETURN                                      <114>
1590 IF WE=141 OR WE=142 OR WE=140 THEN 16
     60                                          <064>
1600 RETURN                                      <134>
1610 PRINT"{CLR,2DOWN}UEBERSETZUNG NICHT{D
     OWN}                                        <110>
1620 PRINT"MOEGLICH, DA ENT-{DOWN}               <035>
1630 PRINT"SPRECHENDE ROUTINE{DOWN}              <225>
1640 PRINT"BEIM VC-20 NICHT VOR-{DOWN}           <096>
1650 PRINT"HANDEN IST !{DOWN}":END               <239>
1660 PRINT"{CLR,2DOWN}UEBERSETZUNG NICHT{D
     OWN}                                        <160>
1670 PRINT"MOEGLICH, DA INS RAM{DOWN}            <098>
1680 PRINT"GEPOKED WIRD{2DOWN}"                  <086>
1690 PRINT"STELLE(DEZ): ";H                      <170>
1700 PRINT"{DOWN}":END                           <074>
1710 REM                                         <248>
1720 REM BEFEHLSLAENGE                           <098>
1730 REM                                         <012>
1740 DATA 0,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <015>
1750 DATA 1,1,1,0,1,1,1,0,0,2,0,0,2,2,2,0        <193>
1760 DATA 2,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <051>
1770 DATA 1,1,1,0,1,1,1,0,0,2,0,0,2,2,2,0        <213>
1780 DATA 0,1,0,0,0,1,1,0,0,1,0,0,2,2,2,0        <047>
1790 DATA 1,1,1,0,0,1,1,0,0,2,0,0,0,2,2,0        <209>
1800 DATA 0,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <077>
1810 DATA 1,1,1,0,1,1,1,0,0,2,0,0,2,2,2,0        <255>
1820 DATA 1,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <105>
1830 DATA 1,1,1,0,1,1,1,0,0,2,0,0,2,2,2,0        <019>
1840 DATA 1,1,1,0,1,1,1,0,0,1,0,0,2,2,2,0        <253>
1850 DATA 1,1,1,0,1,1,1,0,0,2,0,0,2,2,2,0        <039>
1860 DATA 1,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <145>
1870 DATA 1,1,1,0,0,1,1,0,0,2,0,0,0,2,2,0        <035>
1880 DATA 1,1,0,0,1,1,1,0,0,1,0,0,2,2,2,0        <165>
1890 DATA 1,1,1,0,0,1,1,0,0,2,0,0,0,2,2,0        <055>
1900 REM                                         <184>
1910 REM SYSTEMADR.                              <102>
1920 REM                                         <204>
1930 DATA 228,95,228,41,229,0,229,0,229,5,
     229,5,229,10,229,10                         <146>
1940 DATA 229,24,229,24,229,68,229,95,229,
     102,229,129,229,160,229,187                 <156>
1950 DATA 229,180,229,207,229,202,229,229,
     230,50,230,79,230,132,230,184               <067>
1960 DATA 230,182,230,234,232,218,233,33,2
     32,234,233,117,233,200,234,86               <107>
1970 DATA 233,255,234,141,234,28,234,161,2
     34,36,234,178,234,49,234,191                <139>
1980 DATA 234,135,235,30,235,72,235,220,23
     5,121,236,70,235,177,236,94                 <243>
1990 DATA 236,68,237,33,236,120,237,105,23
     6,185,237,228,236,231,237,243               <186>
2000 DATA 236,240,237,254,237,9,238,20,237
     ,12,238,23,237,64,238,228                   <181>
2010 DATA 237,185,238,192,237,199,238,206,
     237,239,238,246,237,254,239,4               <192>
2020 DATA 238,19,239,25,238,179,239,150,23
     8,187,239,163,239,74,240,39                 <100>
2030 DATA 240,20,240,237,240,134,241,79,24
     0,164,241,96,240,189,241,116                <164>
```

Listing. »Reloctable« (Fortsetzung)

```
2040 DATA 241,43,241,224,241,87,242,14,241
     ,202,242,122,242,14,242,199          <233>
2050 DATA 242,80,243,9,242,145,243,74,243,
     15,243,207,243,31,243,223            <083>
2060 DATA 243,47,243,239,243,74,244,10,244
     ,158,245,66,245,175,246,71           <246>
2070 DATA 245,210,246,106,245,221,246,117,
     246,143,247,40,246,155,247,52        <131>
2080 DATA 246,221,247,96,246,228,247,103,2
     46,237,247,112,246,251,247,126       <075>
2090 DATA 247,44,247,175,247,106,247,231,2
     47,208,248,77,247,215,248,84         <086>
2100 DATA 247,234,248,103,248,13,248,138,2
     48,23,248,148,248,46,248,171         <034>
2110 DATA 248,56,248,183,248,65,248,192,24
     8,74,248,201,248,100,248,234         <175>
2120 DATA 248,107,248,234,248,190,249,47,2
     48,225,249,75,249,44,249,142         <021>
2130 DATA 251,151,248,219,251,166,251,234,
     251,205,252,11,252,184,252,246       <254>
2140 DATA 252,202,253,11,252,209,253,17,25
     2,219,253,27,252,226,253,34          <000>
2150 DATA 253,2,253,63,253,16,253,77,253,2
     1,253,82,253,48,253,109              <094>
2160 DATA 253,80,253,141,253,155,253,109,2
     53,249,254,73,254,0,254,80           <090>
2170 DATA 254,7,254,87,254,24,254,102,254,
     28,254,106,254,33,254,246            <146>
2180 DATA 254,37,254,115,254,52,254,130,25
     4,67,254,169,254,194,255,92          <167>
2190 DATA 255,72,255,114,255,129,255,138  <196>
© 64'er
```

Listing. »Reloctable« (Schluß)

Tips & Tricks zum C16

Auch zu Commodores Kleinsten haben unsere Leser ein paar nützliche Tricks entdeckt, die nicht im Handbuch stehen, den Umgang mit diesem Computer aber wesentlich erleichtern.

Wer vom VC 20 oder C 64 auf den C 16/116 umsteigt, wird im ersten Augenblick die bewährte »RESTORE«-Taste vermissen. Hatte sich der Computer einmal »aufgehängt« (zum Beispiel durch eine falsche WAIT-Anweisung oder einen falschen Sprung in ein Maschinenprogramm), so konnte man ihn fast immer mit RUN/STOP-RESTORE zurückholen. Was macht man nun beim C 16/116, wenn sich der Computer in eine Endlosschleife verabschiedet? Nicht gleich ausschalten! Hier gibt es einen kleinen Trick:

Zuerst den »RESET«-Taster drücken (aber noch nicht loslassen). Dann gleichzeitiges Drücken der »RUN/STOP«- und »Commodore«-Taste. Anschließend »RESET«-Taster loslassen und schon befindet man sich im Monitor-Programm, das mit »X« wieder verlassen werden kann. Der alte Speicherinhalt bleibt erhalten!

(C. Q. Spitzner/tr)

Diverses

Der Bildschirm des C 16 beginnt bei $0C00 (= #3072), der zugehörige Farb-RAM bei $0800 (= #2048) und die hochauflösende Grafik (HiRes) bei $2000 (= #8192). Es können also einfarbige Grafiken vom C 64 problemlos übernommen werden. Die Farbintensität für HiRes ist angesiedelt bei $1800(bis $1FFF). Einen Basic-Warmstart erreicht man durch »SYS 32768« ($8000), hier beginnt auch der ROM-Teil. »SYS 62114« (=$F2A4) macht einen kompletten Reset. Wer von anderen Computern gewohnt ist, seinen Monitor durch einen »SYS«-Befehl zu starten, kann dies mit »SYS 62533« (=$F445) tun.

Die ESC-Funktionen lassen sich auch von Maschinensprache aus nutzen (siehe Tabelle). Für das Definieren von Bildschirmfenstern ist es notwendig, die Eckpunkte anzugeben. Hierbei ist $CD = Zeile und $CA = Spalte, also zum Beispiel:

```
LDX #$02
STX $CA      ;dritte Zeile und
STX $CD      ;dritte Spalte
JSR $DE5E    ;»T«-Befehl
LDX #$12
STX $CA      ;18. Zeile und
STX $CD      ;18. Spalte
JMP $DE60    ;»B«-Befehl
```

definiert ein Bildschirmfenster ab dritter Zeile und Spalte mit 15 Zeilen (18 - 3) Länge zu je 15 Zeichen.

Gerade die Möglichkeit, mit »Windows« zu arbeiten, hebt den C 16/116 weit über seine Konkurrenten hervor. Auch die anderen Escape-Funktionen bieten umfangreiche Möglichkeiten der Bildschirmgestaltung.

Übrigens, die »JMP«-Anweisung am Schluß des kleinen Beispielprogramms ist gleichbedeutend mit »JSR« und anschließendem »RTS«.

Tabelle. Die Escape-Funktionen und ihre Adressen:

ESC+	Adresse Dez/Hex	Bedeutung
A	57129/$DF29	Automatisch Einfügen
B	56928/$DE60	Rechte, untere Fensterecke fixieren
C	57126/$DF26	Automatisch Einfügen aufheben
D	56992/$DEA0	Zeile löschen
I	56971/$DE8B	Zeile einfügen
J	57218/$DF82	Cursor an Zeilenanfang setzen
K	57237/$DF95	Cursor an Zeilenende setzen
L	57117/$DF1D	Scrolling-Modus einschalten
M	57120/$DF20	Scrolling-Modus ausschalten
N	55432/$D888	Bildschirm löschen und auf normale Größe schalten
O	56475/$DC9B	Einfüge-, Anführungszeichen-, Reverse- und Blinkmodus ausschalten
P	57057/$DEE1	Zeile löschen
Q	57035/$DECB	Zeile ab Cursor löschen
R	56904/$DE48	Bildschirm löschen und verkleinern
T	56926/$DE5E	Linke, obere Fensterecke fixieren
V	57078/$DEF6	Bildschirm nach oben scrollen
W	57092/$DF04	Bildschirm nach unten scrollen
X	–/–	Escape-Modus auschalten

Weitere Tips

- »WAIT 1,192« wartet darauf, daß eine Taste am Recorder gedrückt wird.
- Der Speicher für die Funktionstasten beginnt bei $0567 und endet bei $05E6.
- Die Einsprungadresse für die USR-Funktion steht in #1281/1282=$0501/0502.

(Frank Plachetta/tr)

Schnelle Hardcopy

Wenn neben dem C16 ein Drucker steht, erwacht auch bald der Wunsch, den Bildschirm direkt über den Drucker auszugeben. »Fast-Hardcopy« schafft das recht flott.

Die Idee zu »Fast-Hardcopy« entstand durch ein Programm im 64'er, Ausgabe 6/85, das für den C16/116 geschrieben war. Dabei wurden mathematische Funktionen auf dem Bildschirm abgebildet und in einer langwierigen Prozedur als Hardcopy zu Papier gebracht. Mit dem Seikosha GP-100 ergab sich dabei eine Zwangspause von sage und schreibe einer halben Stunde. »Fast-Hardcopy« ist vorwiegend in Maschinensprache geschrieben und erledigt diese Aufgabe in 3½ bis 4 Minuten. Mathematische Funktionen werden zwar nicht erzeugt, aber dafür ist egal, was auf dem Bildschirm zu sehen ist – es wird gedruckt (Bild 1).

Da das Programm im Grafikmodus arbeitet, bleiben für Basic nur noch 2 KByte frei. Aus diesem Grund war es nicht möglich, die Maschinenroutine als Datazeilen in das ebenfalls notwendige Basic-Programm einzubinden. Das gesamte Programm besteht also aus zwei Teilen. Der erste Teil ist das Basic-Programm (Listing 1), das Sie ganz gewöhnlich eingeben können. Für den zweiten Teil schalten Sie mit dem Befehl »MONITOR« den TEDMON Ihres C16/116 ein.

Anschließend geben Sie den Befehl »M 1400 147f« ein. Auf dem Bildschirm erscheint der Speicherauszug von $1400 bis $147f als Hexdump. Jetzt tippen Sie Zeile für Zeile die Werte aus Listing 2 ein. Am Schluß jeder Zeile drücken Sie die RETURN-Taste, die Zeile ist dann in den Speicher aufgenommen. Haben Sie dies erledigt, so können Sie sich mit »M 1480 14ff« den nächsten Bereich vornehmen. Nachdem Sie sich auf diese Weise bis zum Schluß (Adresse $1547) des Listings 2 durchgearbeitet haben, gelangen Sie über den Befehl »X« wieder in den Basic-Bereich. Schreiben Sie nun folgenden Befehl:

```
POKE 45,80:POKE 46,21:SAVE"FAST-HC",8
```

Mit diesem kleinen Trick speichern Sie das Basic- und Maschinenprogramm gemeinsam ab, was Sie jetzt auch unbedingt tun sollten.

Nun einiges zur Arbeitsweise des Programms. In den Zeilen 10 und 12 werden zwei Strings definiert. Sie bestehen aus 2 mal 160 Zeichen. Zusammen ergibt das die Anzahl der Bytes, die der Seikosha GP-100 zum Ausdruck einer Bildschirmzeile mit 320 mal 7 Punkten benötigt.

In dem Maschinenprogramm wird jeder Punkt in dieser Zeile umdefiniert, weil der Drucker die Bytes nicht wie den Zeichensatz des Computers waagerecht, sondern senkrecht verarbeitet. Das Maschinenprogramm legt die Bytefolge dann in dem Speicherbereich ab, wo beide Strings liegen (ein Flußdiagramm für das Maschinenprogramm finden Sie im Bild 2). Die Strings werden in Zeile 30 zum Drucker geschickt.

(Jeans Engel/kn)

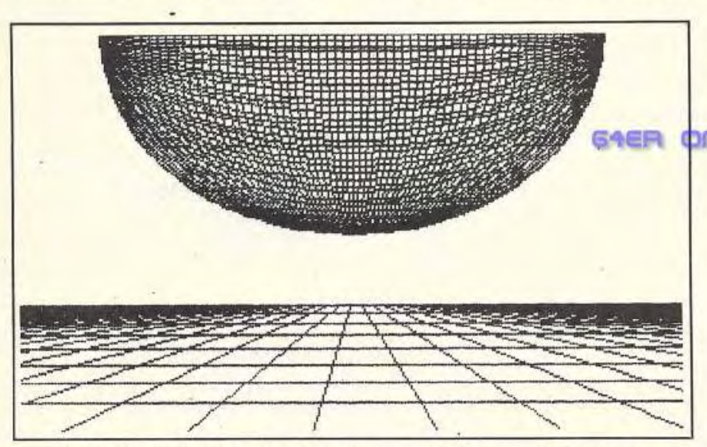


Bild 1. Beispiel für eine Hardcopy mit diesem Programm

```
1 PRINT"{CLR,2DOWN }***{3SPACE,FLASHON}F
AST HARDCOPY{4SPACE}C16/C116{FLASHOFF,4S
PACE}***"
2 PRINT"{DOWN,2SPACE}3.48 MIN{4SPACE}UEB
ER SEIKOSHA GP-100"
3 PRINT"{DOWN }(C)1985 GERMAN SOFT{3SPAC
E}(P) JENS ENGEL"
4 PRINT"{4DOWN }GRAPHIK MUSS SCHON IM SP
EICHER SEIN!"
5 PRINT"{DOWN }DRUCKER FERTIG? <TASTE>"
9 IFPEEK(198)=64THEN9
10 A$="ANFANG****************************
****************************************
***ENDE"
12 X$=A$+A$:Y$=X$:OPEN1,4:GRAPHIC1
18 POKE216,0:POKE217,0:POKE3,0:POKE4,0:P
OKE212,0:FORT=0TO27
30 SYS5242:PRINT#1,CHR$(8);X$;Y$:NEXT
40 SYS5242:FORT=0TO159
42 POKE5900+T,PEEK(5900+T)AND143:POKE573
8+T,PEEK(5738+T)AND143:NEXT
44 PRINT#1,CHR$(8);X$;Y$:GRAPHIC0
©64'er
```

Listing 1. Basic-Programm zu »Fast-Hardcopy«

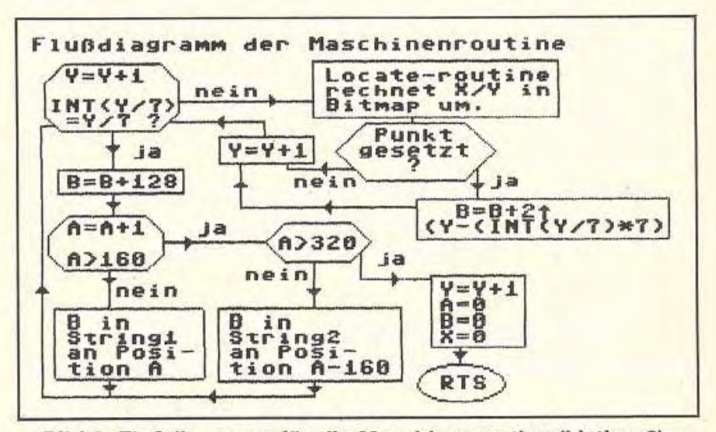


Bild 2. Flußdiagramm für die Maschinenroutine (Listing 2).

```
>1400 A5 D4 29 F8 85 0E 85 05
>1408 A9 00 85 06 06 05 26 06
>1410 06 05 26 06 18 A5 05 65
>1418 0E 85 05 A5 06 69 00 85
>1420 06 06 05 26 06 06 05 26
>1428 06 06 05 26 06 A5 D4 29
>1430 07 18 65 05 85 05 A5 06
>1438 69 00 85 06 18 A5 03 29
>1440 F8 65 05 85 05 A5 04 65
>1448 06 85 06 18 A9 00 65 05
>1450 85 05 A9 20 65 06 85 06
>1458 A5 03 29 07 49 07 AA A9
>1460 01 CA 30 03 0A D0 FA A0
>1468 00 31 05 F0 07 A9 01 60
>1470 EA 60 EA EA A9 00 60 EA
>1478 60 EA A9 00 85 D3 EA EA
>1480 20 00 14 F0 06 A9 01 05
>1488 D3 85 D3 E6 D4 20 00 14
>1490 F0 06 A9 02 05 D3 85 D3
>1498 E6 D4 20 00 14 F0 06 A9
>14A0 04 05 D3 85 D3 E6 D4 20
>14A8 00 14 F0 06 A9 08 05 D3
>14B0 85 D3 E6 D4 20 00 14 F0
>14B8 06 A9 10 05 D3 85 D3 E6
>14C0 D4 20 00 14 F0 06 A9 20
>14C8 05 D3 85 D3 E6 D4 20 00
>14D0 14 F0 06 A9 40 05 D3 85
>14D8 D3 A9 80 05 D3 85 D3 E6
>14E0 03 D0 02 E6 04 C6 D4 C6
>14E8 D4 C6 D4 C6 D4 C6 D4 C6
>14F0 D4 EA EA EA EA EA EA A5
>14F8 D8 C9 A0 F0 13 A4 D8 E6
>1500 D8 A5 D3 99 0C 17 4C 7A
>1508 14 EA EA EA EA EA EA EA
>1510 A5 D9 C9 A0 F0 0F A4 D9
>1518 E6 D9 A5 D3 99 6A 16 4C
>1520 7A 14 EA EA EA EA EA EA
>1528 EA EA EA EA EA EA A9 00
>1530 85 03 85 04 A5 D4 18 69
>1538 07 85 D4 A9 00 85 D8 85
>1540 D9 60 00 00 00 00 00 00
```

Listing 2. Maschinenprogramm zu »Fast-Hardcopy«. Bitte mit dem eingebauten Monitor eingeben.

Super-Assembler mit Befehlserweiterung

Endlich ein komfortabler Assembler für den C16. Die Zeiten der simplen Monitorprogrammierung sind vorbei. Jetzt können Sie Maschinenprogramme mit Label schreiben, Texte im ASCII-Code einfügen, dezimale oder duale Daten eingeben und vieles mehr.

Haben Sie auch schon vor dem Computer gesessen und sich bei der Eingabe von Maschinenprogrammen über den unkomfortablen Monitor beim C16 geärgert? Wir wollen da Abhilfe schaffen. Der Assembler (Listing 2) arbeitet mit Label und bietet Ihnen die Möglichkeit, einen Quelltext zu erzeugen, bei dem Sie Daten wahlweise im Dual-, Dezimal-, Hex- oder ASCII-Code eingeben. Vom Ladeprogramm (Listing 1) wird eine sinnvolle Belegung der Funktionstasten generiert.

Nun näheres zum Assembler. Der Quelltext wird mit Hilfe des Basic-Editors eingegeben, ähnlich wie ein Basic-Programm. Das erste Zeichen in einer Zeile, die assembliert werden soll, muß ein Doppelpunkt sein. Vor einem Mnemonic ist mindestens ein Leerzeichen zu setzen. Label dürfen eine beliebige Länge haben, müssen aber direkt auf den Doppelpunkt folgen. Zum Einfügen von Daten und Text in den Quellcode stehen verschiedene Pseudobefehle zur Verfügung (siehe Tabelle 1), denen zur Erkennung ein Punkt vorangestellt wird. Vor diese kann ein Label gesetzt werden, allerdings müssen Sie hier ebenfalls ein Leerzeichen als Trennung einfügen. Daten können Dezimal, Hexadezimal (ein $ vorangestellt), Dual (%) und als ASCII-Codes (in Anführungszeichen) eingegeben werden. Als arithmetische Operationen stehen Additionen und Subtraktionen zur Verfügung. Normale Basic-Programme können Sie wie bisher editieren, speichern, laden und starten.

Nach der Zeilennummer müssen Sie nicht etwa einen Doppelpunkt und eine Reihe Leertasten drücken, es genügt die Taste F1. Statt ":." betätigen Sie nur F8 (HELP).

Was der Assembler alles bietet

Mit F4 und »RETURN« starten Sie die Assemblierung. Die meisten Fehler werden vom Programm erkannt. Der Computer wirft dann die Meldung »SYNTAX ERROR« und die entsprechende Zeilennummer aus.

Um ein Programm von Diskette zu laden, brauchen Sie nur mit den Cursortasten auf den betreffenden Eintrag im Directory fahren und die Funktionstaste F2 drücken. Den Quellcode kann man am besten speichern, indem man die Taste F5 betätigt und der Name des Programms eingegeben wird. Eine eventuell vorhandene ältere Version mit gleichem Namen wird dabei überschrieben.

Im Directory der Diskette finden Sie den Programmnamen mit dem angehängten Kürzel ».ASS« wieder. Der Assembler belegt 2896 Bytes am Anfang des Speichers. Der Speicher für Quelltext und Labeltabelle beginnt jetzt bei $1B50. 20 KByte Quelltext werden in zirka 18,5 Sekunden assembliert. Davon muß man noch die Zeit für die Interrupts abrechnen, die normalerweise abgeschaltet werden. Eine eventuell vorhandene Speichererweiterung kann vom Programm voll ausgenutzt werden.

(Thomas Tieke/kn)

Pseudobefehle

Befehl	Beschreibung	Zeile	Label	Operand
.ac	– Fügt ASCII-Texte in den Quelltext ein.			
	Beispiel:	100 :.BA=$7000		
		110 ————		
		120 :		JMP MAIN
		130 :TEXT		.AC »Hallo, Sie da vorne!!«
		140 :		.BY 13
		150 ————		
		160 :MAIN		LDX #»A«
		170 ...		
.ba	– Startadresse des Maschinenprogramms festlegen			
	Beispiel:	100 ————		
		110 :.BA=		$2000
		120 :.EQ OUT=		$FFD2
		130 :.EQ IN=		$FFE4
		140 ...		
.by	– Fügt Daten in den Quelltext ein			
	Beispiel:	100 :.BA = $6000		
		110 :LABEL		.BY TEST,0,$FFFF,%1110+$44
		120 :		.BY $10,$10,$10FF-LABEL
		130 :TEXT		LDA LABEL, X
		140 :		STA TEST, Y
		150 ...		
.eq	– Weist einem Label einen Wert zu.			
	Beispiel:	100 ...		
		110 :		.BA= BASIS+TEXT
		120 :.EQ AUSGABE = $FFD2		
		130 :.EQ EINGABE = AUSGABE-555		
		140 ————		
		150 :MAIN		LDA # 0
		160 :		TAX
		170 ...		

Tabelle. Befehlserweiterung von »ASSEMBLER«

```
100 KEY5,"DSAVE"+CHR$(34)+"@{12SPACE}.AS
S"+CHR$(34)+"{17LEFT}"
110 KEY1,":{10SPACE}"
120 KEY8,":."
130 KEY4,"SYS6780"
140 KEY2,"DLOAD"+"{18RIGHT}:{19LEFT}"+CH
R$(34)+CHR$(13)
150 F$=CHR$(34)
160 A$="POKE43,DEC("+F$+"50"+F$+"):"
170 PRINT"{CLR}POKE44,DEC("+F$+"1B"+F$+"
):POKEDEC("+F$+"1B4F"+F$+"),0:"+A$+"NEW"
180 PRINT"{2DOWN}LOAD"+F$+"ASSEMBLER"+F$
+",8,1"
190 PRINT"{4DOWN}NEW"
200 POKE1319,19:POKE1320,13:POKE1321,13:
POKE1322,13:POKE239,4
```

Listing 1. Ladeprogramm für »ASSEMBLER«

```
MONITOR

   PC  SR AC XR YR SP
; .FF00 00 00 FF 00 F8

>1000 20 2F 18 C9 20 F0 F9 C9
>1008 23 D0 05 E6 D8 4C 00 10
>1010 C9 28 D0 05 E6 D9 4C 00
>1018 10 A2 00 86 14 20 D3 19
>1020 EA D0 03 4C B1 12 C9 24
>1028 D0 03 4C 18 12 C9 22 D0
>1030 03 4C AA 12 4C 43 12 D0
>1038 02 AF 10 38 E9 30 C9 0A
>1040 90 01 60 26 14 26 15 A6
>1048 14 86 03 A6 15 86 04 18
>1050 26 14 26 15 18 26 14 26
>1058 15 AA A5 14 18 65 03 85
>1060 14 A5 15 65 04 85 15 8A
>1068 18 65 14 85 14 90 02 E6
>1070 15 C8 B1 3B 4C 3B 10 C8
>1078 B1 3B 38 E9 30 C9 0A 90
>1080 07 E9 07 C9 10 90 01 60
>1088 A2 00 18 26 14 26 15 E8
>1090 E0 04 D0 F6 05 14 85 14
>1098 4C 77 10 C8 B1 3B 38 E9
>10A0 30 C9 02 90 01 60 C9 01
>10A8 26 14 26 15 4C 9B 10 C8
>10B0 B1 3B 85 14 C8 C8 60 A5
>10B8 2D 85 D0 A5 2E 85 D1 C8
>10C0 98 48 84 D7 A0 00 B1 D0
>10C8 A4 D7 C9 FF 4C DB 17 D1
>10D0 3B D0 07 20 14 11 E6 D7
>10D8 D0 EA 4C CB 1A 20 14 11
>10E0 A0 00 B1 D0 C9 3B D0 F5
>10E8 20 14 11 68 85 D7 48 4C
>10F0 C4 10 4C C8 10 A0 00 C8
>10F8 B1 D0 85 14 C8 B1 D0 85
>1100 15 68 A4 D7 60 68 A9 00
>1108 85 14 4C 6F 19 E6 3B D0
>1110 02 E6 3C 60 E6 D0 D0 02
>1118 E6 D1 60 A5 2B 85 3B A5
>1120 2C 85 3C A5 2D 85 D0 A5
>1128 2E 85 D1 A9 00 85 D4 85
>1130 D5 20 B0 1A EA 85 D2 20
>1138 0D 11 B1 3B 85 D3 A5 D2
>1140 D0 09 A5 D3 D0 05 A9 FF
>1148 91 D0 60 20 0D 11 20 0D
>1150 11 20 12 12 C9 3A D0 16
>1158 20 0D 11 B1 3B C9 20 D0
>1160 03 4C 6A 1A C9 2E D0 03
>1168 4C 79 11 4C 9F 11 A5 D2
>1170 85 3B A5 D3 85 3C 4C 31
>1178 11 20 A3 19 C9 42 D0 EE
>1180 C8 B1 3B C9 41 D0 E7 C8
>1188 B1 3B C9 B2 F0 03 4C A5
>1190 1A 20 8D 18 A5 14 85 D4
>1198 A5 15 85 D5 4C 6E 11 A5
>11A0 D0 48 A5 D1 48 20 E8 15
>11A8 E0 99 F0 07 4C A5 1A 03
>11B0 EA E6 3B 68 85 D1 68 85
>11B8 D0 A0 00 B1 3B F0 ED 20
>11C0 8D 1A 90 0C EA 91 D0 20
>11C8 0D 11 20 14 11 4C B9 11
>11D0 A9 3D 91 D0 20 14 11 A5
>11D8 D4 91 D0 20 14 11 A5 D5
>11E0 91 D0 20 14 11 A9 3B 91
>11E8 D0 4C 40 1A A2 01 B1 3B
>11F0 C8 C9 20 F0 F9 20 7F 18
>11F8 EA F0 0A 20 8D 18 A2 02
>1200 20 F5 13 EA EA 8A 18 65
>1208 D4 85 D4 90 02 E6 D5 4C
>1210 6E 11 20 0D 11 B1 3B 60
>1218 84 DD A2 00 C8 B1 3B C9
>1220 30 90 0F C9 3A B0 03 E8
>1228 D0 F2 C9 41 90 04 C9 47
>1230 90 F5 E0 03 90 04 A9 FF
>1238 D0 02 A9 00 20 AD 13 A8
>1240 4C 77 10 C9 25 D0 23 84
>1248 DD A2 00 C8 B1 3B C9 30
>1250 90 07 C9 33 B0 03 E8 D0
>1258 F2 E0 09 90 04 A9 FF D0
>1260 02 A9 00 20 AD 13 A8 4C
>1268 9B 10 84 DD A2 00 C8 B1
>1270 3B C9 30 90 07 C9 3A B0
>1278 03 E8 D0 F2 E0 03 90 21
>1280 D0 1B 88 88 88 B1 3B C9
>1288 32 90 16 D0 10 C8 B1 3B
>1290 C9 35 90 0D D0 07 C8 B1
>1298 3B C9 36 90 04 A9 FF D0
>12A0 02 A9 00 20 AD 13 A8 4C
>12A8 72 10 A9 00 85 DB 4C AF
>12B0 10 A9 FF 85 DB 4C B4 13
>12B8 84 DC 4C 9F 11 84 DC 4C
>12C0 EC 11 20 DE 12 A5 DB C9
>12C8 07 90 05 C9 0B B0 01 E8
>12D0 60 A9 00 85 D8 85 D9 85
>12D8 E1 85 EA 4C 4F 18 A2 00
>12E0 B5 D8 C9 02 90 03 4C A5
>12E8 1A E8 E0 02 D0 F2 A5 D8
>12F0 F0 0D C5 D9 F0 F0 A5 DB
>12F8 D0 EC A9 0B 85 DB 60 A5
>1300 DB C9 80 D0 04 A9 01 D0
>1308 F3 C9 FF D0 34 A5 D9 F0
>1310 04 A9 0A D0 E7 A4 DC B1
>1318 3B F0 17 C9 2C D0 C7 C8
>1320 B1 3B C9 58 D0 04 A9 08
>1328 D0 D2 C9 59 D0 B8 A9 09
>1330 D0 CA 20 CA 13 E0 08 90
>1338 04 A9 07 D0 BF A9 0C D0
>1340 BB A5 D9 D0 21 A4 DC B1
>1348 3B D0 04 A9 02 D0 AD C9
>1350 2C D0 93 C8 B1 3B C9 58
>1358 D0 04 A9 03 D0 9E C9 59
>1360 D0 84 A9 04 D0 96 A4 DC
>1368 A2 FF EA EA EA EA EA E8
>1370 B1 3B DD 9B 13 D0 0A 20
>1378 D3 17 D0 F3 A9 05 85 DB
>1380 60 E0 00 F0 03 4C A5 1A
>1388 A2 FF E8 B1 3B DD 9E 13
>1390 D0 F3 20 D3 17 D0 F3 A9
>1398 06 D0 E3 2C 58 29 29 2C
>13A0 59 A5 D0 48 A5 D1 48 4C
>13A8 E5 13 60 DC 60 85 DB 84
>13B0 DC A5 DD 60 84 DD C8 B1
>13B8 3B 20 8D 1A 90 05 EA EA
>13C0 EA D0 F3 84 DC A4 DD 4C
>13C8 B7 10 98 48 88 B1 3B C9
>13D0 20 D0 F9 88 B1 3B C9 20
>13D8 F0 F9 88 88 20 EC 17 68
>13E0 A8 60 A2 09 60 A9 13 48
>13E8 A9 ED 48 4C 00 10 68 85
>13F0 D1 68 4C 58 18 20 E0 17
>13F8 A2 02 A5 DB C9 07 90 05
>1400 C9 0B B0 01 E8 60 42 43
>1408 43 90 42 43 53 B0 42 45
>1410 51 F0 42 4E 45 D0 42 4D
>1418 49 30 42 50 4C 10 42 56
>1420 43 50 42 56 53 70 42 52
>1428 4B 00 52 54 49 40 52 54
>1430 53 60 50 48 50 08 43 4C
>1438 43 18 50 4C 50 28 53 45
>1440 43 38 50 48 41 48 43 4C
>1448 49 58 50 4C 41 68 53 45
>1450 49 78 44 45 59 88 54 59
>1458 41 98 54 41 59 A8 43 4C
>1460 56 B8 49 4E 59 C8 43 4C
>1468 44 D8 49 4E 58 E8 53 45
>1470 44 F8 54 58 41 8A 54 58
>1478 53 9A 54 41 58 AA 54 53
>1480 58 BA 44 45 58 CA 4E 4F
>1488 50 EA 41 53 4C 0A 4C 53
>1490 52 4A 52 4F 4C 2A 52 4F
>1498 52 6A 4A 4D 53 52 20 4A 4D
>14A0 50 4C 43 50 58 E0 43 50
>14A8 59 C0 4C 44 58 A2 4C 44
>14B0 59 A0 53 54 58 8E 53 54
>14B8 59 8C 42 49 54 2C 53 54
>14C0 41 8D 44 45 43 CE 49 4E
>14C8 43 EE 41 44 43 6D 41 4E
>14D0 44 2D 43 4D 50 CD 45 4F
>14D8 52 4D 4C 44 41 AD 4F 52
>14E0 41 0D 53 42 43 ED A2 00
>14E8 B1 3B DD 06 14 D0 0F E8
>14F0 C8 8A 29 03 49 03 D0 F0
>14F8 CA 8A 4A 4A AA 60 8A 29
>1500 03 F0 05 CA 88 4C FE 14
>1508 E8 E8 E8 E8 E0 E0 90 D8
>1510 A2 FF 60 20 1B 11 A5 2B
>1518 85 3B A5 2C 85 3C A5 2D
>1520 85 D0 A5 2E 85 D1 A9 00
>1528 85 D4 85 D5 20 B0 1A EA
>1530 85 D2 C8 B1 3B 85 D3 A5
>1538 D2 D0 05 A5 D3 D0 01 60
>1540 C8 C8 C8 B1 3B C9 3A D0
>1548 3D C8 B1 3B C9 2E D0 03
>1550 4C 65 1A C9 20 F0 07 C8
>1558 B1 3B C9 20 D0 F9 C8 B1
>1560 3B C9 20 F0 F9 4C 56 1A
>1568 E0 FF D0 03 4C A5 1A 86
>1570 DE E8 8A 0A 0A AA CA BD
>1578 06 14 85 DF 88 20 C5 15
>1580 20 DE 12 4C 15 16 A5 D2
>1588 85 3B A5 D3 85 3C 4C 2C
>1590 15 A2 FF C8 E8 B1 3B DD
>1598 CC 15 F0 F7 BD CC 15 C9
>15A0 FF D0 09 BD CD 15 48 BD
>15A8 CE 15 48 60 E8 BD CC 15
>15B0 C9 FF D0 F8 A4 DD E8 BD
>15B8 CE 15 C9 FF F0 AE EA EA
>15C0 EA E8 4C 93 15 20 8D 18
>15C8 4C F5 15 15 42 41 B2 FF
>15D0 15 FF 45 51 20 FF 19 41
>15D8 42 59 FF 19 7C 41 43 FF
>15E0 1A 15 FF FF FF FF FF FF
>15E8 A2 FF A5 3B D0 02 C6 3C
>15F0 C6 3B 4C B7 10 A5 EA C9
>15F8 FF D0 02 F0 43 60 EA EA
>1600 20 CE 17 A5 14 85 D4 A5
>1608 15 85 D5 4C 86 15 E6 D4
>1610 D0 02 E6 D5 60 A0 00 A2
>1618 00 A5 DE C9 08 B0 17 A5
>1620 DF 91 D4 20 0E 16 A5 14
>1628 38 20 00 1B CA BA 91 D4
>1630 20 0E 16 4C 86 15 C9 21
>1638 B0 15 A5 DB C9 01 F0 03
>1640 4C A5 1A A5 DF 4C 2E 16
>1648 A5 DF 91 D4 4C 0E 16 C9
>1650 25 B0 45 A5 DB C9 01 F0
>1658 EA C9 03 D0 0D A5 DF 18
>1660 69 0C 20 4A 16 A5 14 4C
>1668 2E 16 C9 08 D0 12 A5 DF
>1670 18 69 14 20 4A 16 A5 14
>1678 20 4A 16 A5 15 4C 2E 16
>1680 C9 07 D0 08 A5 DF 18 69
>1688 04 4C 73 16 C9 02 D0 B0
>1690 A5 DF 38 E9 04 4C 62 16
>1698 C9 25 D0 0B A5 DB C9 07
>16A0 D0 9E A5 DF 4C 73 16 C9
>16A8 26 D0 12 A5 DB C9 07 F0
>16B0 F1 C9 0A D0 BB A5 DF 18
>16B8 69 20 4C 73 16 C9 29 B0
>16C0 26 A5 DB C9 0B D0 05 A5
>16C8 DF 4C 62 16 C9 02 D0 08
>16D0 A5 DF 18 69 04 4C 62 16
>16D8 C9 07 F0 03 4C A5 1A A5
>16E0 DF 18 69 0C 4C 73 16 C9
>16E8 29 D0 26 A5 DB C9 0B F0
>16F0 D6 C9 02 F0 DB C9 07 F0
>16F8 E6 C9 04 D0 08 A5 DF 18
>1700 69 14 4C 62 16 C9 09 D0
>1708 D3 A5 DF 18 69 1C 4C 73
>1710 16 C9 2A D0 0C A5 DB C9
>1718 08 F0 EE C9 03 F0 DE D0
>1720 CC C9 2D B0 27 A5 DB C9
>1728 07 D0 05 A5 DF 4C 73 16
>1730 C9 02 D0 08 A5 DF 38 E9
>1738 08 4C 62 16 C9 04 F0 04
>1740 C9 03 D0 98 A5 DF 18 69
>1748 08 4C 62 16 C9 2D D0 0C
>1750 A5 DB C9 07 F0 D5 C9 02
>1758 F0 DA D0 80 C9 2E D0 37
>1760 A5 DB C9 07 F0 C5 C9 02
>1768 F0 CA C9 09 D0 03 4C DF
>1770 16 C9 06 D0 03 4C D0 16
>1778 C9 03 F0 C8 C9 08 D0 08
>1780 A5 DF 18 69 10 4C 73 16
>1788 C9 05 F0 03 4C DC 16 A5
>1790 DF 38 E9 0C 4C 62 16 C9
>1798 31 B0 14 A5 DB C9 07 F0
>17A0 8A C9 02 F0 8F C9 08 F0
```

Listing 2. »ASSEMBLER«. Bitte mit dem eingebauten Monitor eingeben.

```
>17A8 D7 C9 03 F0 97 D0 DD A5
>17B0 DB C9 0B D0 AD A5 DF 38
>17B8 E9 04 4C 62 16 C8 B1 3B
>17C0 F0 01 60 85 14 85 15 A9
>17C8 80 85 DB 60 68 60 88 20
>17D0 8D 18 60 C8 E0 02 60 EA
>17D8 C9 03 60 85 EA 4C EB 1A
>17E0 4C DE 12 A9 FF A0 00 91
>17E8 2D 4C 13 15 98 48 B1 3B
>17F0 C9 D1 F0 0F C9 45 F0 15
>17F8 C9 B0 F0 22 C9 AF F0 08
>1800 4C 39 1B C8 68 A2 2F 60
>1808 C8 68 A2 32 60 C8 B1 3B
>1810 C9 B0 F0 05 68 A8 4C E6
>1818 14 C8 68 A2 34 60 C8 B1
>1820 3B C9 41 D0 EF C8 68 A2
>1828 36 60 68 A8 4C E6 14 20
>1830 BD 17 C9 80 D0 03 68 68
>1838 60 C9 B3 D0 07 E6 E0 68
>1840 68 4C 00 10 C9 B1 D0 F0
>1848 E6 E1 68 68 4C 00 10 85
>1850 E0 A9 80 85 DB 4C A1 13
>1858 85 D0 C6 E0 F0 05 C6 E1
>1860 F0 08 60 A9 00 85 DB 85
>1868 15 60 A5 15 85 14 A9 00
>1870 85 15 85 DB 60 20 14 11
>1878 A9 FF 91 D0 60 88 60 B1
>1880 3B F0 07 C9 20 F0 03 C8
>1888 D0 F5 B1 3B 60 20 D1 12
>1890 B1 3B C9 AA F0 05 C9 AB
>1898 F0 01 60 85 E2 A5 14 85
>18A0 E3 A5 15 85 E4 A5 D8 85
>18A8 E5 A5 D9 85 E6 A5 E0 85
>18B0 E7 A5 E1 85 E8 A5 DB 85
>18B8 E9 20 D1 12 A5 E2 C9 AA
>18C0 D0 10 A5 E3 18 65 14 85
>18C8 14 A5 E4 65 15 85 15 4C
>18D0 DF 18 A5 E3 38 E5 14 85
>18D8 14 A5 E4 E5 15 85 15 A5
>18E0 E5 85 D8 A5 E6 85 D9 A5
>18E8 E7 85 E0 A5 E8 85 E1 A5
>18F0 E9 85 DB 60 B1 3B C8 C9
>18F8 20 F0 F9 88 20 8D 1A 90
>1900 16 EA EA EA AA 98 48 8A
>1908 A0 00 91 D0 68 A8 20 0D
>1910 11 20 14 11 4C F4 18 C9
>1918 B2 F0 03 4C A5 1A 20 8D
>1920 18 A0 00 A9 3D 91 D0 20
>1928 14 11 A5 14 91 D0 20 14
>1930 11 A5 15 91 D0 20 14 11
>1938 A9 3B 91 D0 20 14 11 4C
>1940 68 19 98 48 A0 00 A5 2D
>1948 85 D0 A5 2E 85 D1 B1 D0
>1950 C9 3B F0 06 20 14 11 4C
>1958 4E 19 20 14 11 B1 D0 C9
>1960 FF D0 EB 68 A8 4C F4 18
>1968 A9 FF 91 D0 4C 86 15 85
>1970 15 C8 B1 3B 20 8D 1A 90
>1978 03 B0 F6 F5 60 20 8D 18
>1980 84 DD A5 14 A0 00 91 D4
>1988 20 0E 16 A5 DB C9 FF D0
>1990 07 A5 15 91 D4 20 0E 16
>1998 A4 DD B1 3B C9 2C F0 DD
>19A0 4C 86 15 84 DD C8 B1 3B
>19A8 C9 42 D0 3F EA C8 B1 3B
>19B0 C9 59 F0 05 4C E4 19 83
>19B8 11 20 8D 18 20 0E 16 A5
>19C0 DB C9 FF D0 03 20 0E 16
>19C8 B1 3B C9 2C F0 EB 68 68
>19D0 4C 6E 11 86 15 20 8D 1A
>19D8 90 07 EA EA EA 88 A2 00
>19E0 60 A2 FF 60 AA 68 68 8A
>19E8 4C 83 11 C9 41 F0 01 60
>19F0 C8 B1 3B C9 43 D0 D7 C8
>19F8 B1 3B C9 22 D0 F9 A2 00
>1A00 C8 E8 B1 3B C9 22 D0 F8
>1A08 CA 8A 18 65 D4 85 D4 90
>1A10 02 E6 D5 4C CE 19 B1 3B
>1A18 C9 20 D0 03 C8 D0 F7 C9
>1A20 22 F0 03 4C A5 1A C8 B1
>1A28 3B C9 22 F0 10 AA 98 48
>1A30 8A A0 00 91 D4 20 0E 16
>1A38 68 A8 4C 26 1A 4C 86 15
>1A40 20 75 18 B1 3B C9 20 D0
>1A48 03 C8 D0 F7 C9 2E F0 03
>1A50 4C EC 11 4C 79 11 C9 2E
>1A58 D0 05 84 DD 4C 91 15 20
>1A60 EC 17 4C 68 15 84 DD 4C
>1A68 91 15 B1 3B C8 C9 20 F0
>1A70 F9 88 C9 2E F0 03 4C EC
>1A78 11 4C 79 11 78 A9 FF 8D
>1A80 3F FF 20 E3 17 AD 3F FF
>1A88 8D 3E FF 58 60 C9 41 90
>1A90 10 C9 AA 90 0E C9 B4 B0
>1A98 0A C9 AF F0 06 C9 B0 F0
>1AA0 02 18 60 38 60 A9 FF 8D
>1AA8 3E FF 58 A2 0B 4C 86 86
>1AB0 A0 02 B1 3B 85 39 C8 B1
>1AB8 3B 85 3A A0 00 B1 3B 60
>1AC0 EA EA EA EA EA EA EA EA
>1AC8 EA EA EA C9 3D F0 03 4C
>1AD0 DD 10 B1 3B C9 40 90 10
>1AD8 C9 60 90 F3 C9 80 90 08
>1AE0 C9 AA 90 EB C9 AF B0 E7
>1AE8 4C F5 10 F0 03 4C CF 10
>1AF0 E0 FF D0 08 A2 99 E6 3B
>1AF8 D0 02 E6 3C 4C 05 11 00
>1B00 E5 D4 AA A5 D5 85 E7 CA
>1B08 30 1B E8 8A 18 65 D4 85
>1B10 E6 90 02 E6 E7 A5 E7 C5
>1B18 15 F0 03 4C A5 1A A5 E6
>1B20 C5 14 D0 F7 60 8A 49 FF
>1B28 E8 85 E6 A5 D4 38 E5 E6
>1B30 85 E6 B0 02 C6 E7 4C 15
>1B38 1B C9 52 F0 03 4C 2A 18
>1B40 C8 B1 3B C9 B0 F0 03 4C
>1B48 14 18 C8 68 A2 24 60 00
>1B50 00 00 00 00 00 00 00 00
```


Listing 2. »ASSEMBLER« (Schluß)

16 Farben für den VC 20

Welcher VC 20-Besitzer träumt nicht davon, mit seinem Computer 16 Farben darzustellen?

Nun ist auch der VC 20 in der Grundversion imstande, wie sein großer Bruder, der C 64, eine Palette von 16 Farben anzubieten. Das Programm gebraucht den sogenannten »Extend Color Mode«, den viele gar nicht kennen, mit dem aber Einiges getan werden kann. Das Listing schreibt eine kurze Maschinenroutine in einen von Basic nicht benutzten Speicherbereich. Die Syntax des neuen Kommandos lautet: !FARBE BORDER (0 - 7), TEXT (0 - 15). Dabei bedeuten:

0	schwarz	8	orange
1	weiß	9	hellorange
2	rot	10	rosa
3	Cyan	11	hellcyan
4	violett	12	hellviolett
5	grün	13	hellgrün
6	blau	14	hellblau
7	gelb	15	hellgelb

(Marcel Kreeft/dm)

```
10 PRINT"{CLR}MOMENT"                         <031>
20 CK=0                                       <172>
30 FOR G=673 TO 754:READ A:POKE G,A:CK=CK+
   A:NEXT                                     <209>
40 IF CK<>8662 THEN PRINT"DATA ERROR":END    <194>
50 DATA 169,172,141,008,003,169,002,141,00
   9,003,096,032,115,000,201,033,240         <059>
60 DATA 009,032,121,000,076,231,199,076,00
   8,207,160,000,032,115,000,217,238         <114>
70 DATA 002,208,243,200,192,005,208,243,03
   2,155,215,142,132,003,032,121,000         <255>
80 DATA 201,044,208,225,032,155,215,138,01
   0,010,010,010,109,132,003,141,015         <100>
90 DATA 144,169,005,032,210,255,076,174,19
   9,070,065,082,066,069                     <045>
100 SYS 673                                   <104>
110 !FARBE 6,14                               <167>
120 PRINT"{CLR,6SPACE,RVSON}16 FARBEN"        <153>
130 PRINT"{3DOWN}SYNTAX: !FARBE BORDER(0-7
    ), TEXT(0-15)"                            <248>
140 PRINT"{RVSON,38SPACE}"                    <104>
150 PRINT"{DOWN,16SPACE}TASTE                 <127>
160 PRINT"{DOWN,RVSON,38SPACE}"               <242>
170 GET Z$:IF Z$=""THEN 170                   <065>
180 PRINT"{CLR}";:!FARBE 3,6                  <187>
190 PRINT"********!FARBE********":NEW        <209>
```


Listing. »16 Farben« mit dem VC 20

In die Datei geschaut

Wollen Sie in Dateien des C16 herumschnüffeln oder sie gar ändern, dann haben wir hier das richtige Programm für Sie.

Mit dem Hilfsprogramm »P++Z« (Listing) können Sie alle sequentiellen Files bearbeiten, die sich mit »INPUT #« lesen lassen. Grundideen für dieses Programm sind:
- namentlich bekannte Dateien lesen, ohne das zuständige Anwenderprogramm zu benutzen,
- Dateien kopieren (Sicherheitskopien).

Sie können aber auch die einzelnen Variablen einer Datei verändern oder löschen. Beim Verändern müssen Sie darauf achten, daß der vorhandene Variablentyp erhalten bleibt. Wenn Sie nicht wissen, ob eine vorhandene Zahl vom zuständigen Anwenderprogramm als String oder Integervariable gelesen wird, dann sollten Sie Zahlen auch nur durch andere Zahlen ersetzen. Eine besondere Leistung des Programms ist die Absicherung gegen ein versehentliches Überschreiben von Daten. Vor dem Überschreiben einer Datei wird diese in »UTHELP« umbenannt und gespeichert. Die Umbe-

Variable	Bedeutung / Zeilen
A$	File-Variable 340 410 1090 1200 1260 1300 5440 5450
EM$	Error-Meldungen 25 29 5120
EN	Error-Nummern 25 27 28 5120 5130 5140
ES	Error-Trak 25 5120
ET	Error-Sektor 25 5120
GF$	Anfangs-Zeichen 195 1090 1200
GS	Gesamtzahl Variablen 340 350 1000 1080 1160 5430
I	Indiz 280 320 340 400 410 440
KT	Änderungs-Flag 270 1310 5400 9700
L	Loop 1160 1190 1200 5430 5440 5450 9730
N1$	Aktuelle Datei 210 220 260 380 5070 5350
N2$	SAVE Datei 5070 5080 5090 5310 5410 5420
N3$	RVS + Dateiname 220 230 1000 1150 1100 5300
Q$	Hilfsvariable 27 28 270 290 390 1040 1050 1060 1070 1080 1100 1110 1180 1220 5050 9710
Q%	Index 1080 1090 1260 1300
S1$	Strich 194 1010 1030
S2$	Sternenreihe 193 1090
ST	Status 300 310 420 430
X$	Hilfsvariable 5000 5010

Tabelle 1. Variablenliste zu »P++Z«

Zeile	Bemerkung
1	Programmname, letzte Änderung, Programm-Länge
19	Sprung nach 190, da am Programmanfang oft benutzte Unterprogramme gut aufgehoben sind.
24- 29	Floppy Error Unterprogramm
190- 195	Standardvariable (Programmanfang)
200- 230	N1$ und N3$ erhalten ihren Inhalt (siehe Variablenliste)
250- 350	Die Anzahl der Variablen wird für die DIM-Anweisung ermittelt. Dieser Teil sichert auch beim kleinen Speicherplatz des C 16 eine optimale Ausnutzung. Bei mehr als 30 Variablen wird deren Anzahl angezeigt.
370- 460	Die Variablen werden eingelesen.
1000-1330	Hier steht das Hauptprogramm. Zuerst gibt man die laufende Nummer der gewünschten Variable an. Diese wird gezeigt und man kann nach Eingabe von »A« diese ändern, beziehungsweise die Variable wird nach Eingabe von »L« gelöscht. Jede andere Eingabe führt das Programm wieder zur Zeile 1000. Ein »A« anstelle der laufenden Nummer läßt alle eingelesenen Variablen auf dem Bildschirm erscheinen. Hier sorgt Zeile 1170, daß, nachdem der Bildschirm einmal gefüllt wurde, eine Programmpause bis zum Drücken einer Taste eingelegt wird.
1240-1250	zeigen eine für Basic 3.5 spezielle Möglichkeit, eine Bildschirmzeile zu löschen. Zuerst muß der Cursor eine Zeile nach oben (dort steht »Welche Nummer ? xx«). Jetzt wird ein CHR$ (27) in den Bildschirm geschrieben. C 16-Besitzer sollten wissen, wenn dem CHR$(27) ein »Q« folgt, wird ab Cursorposition die Zeile gelöscht, ohne daß sich dabei die Cursorposition selbst verändert. Zeile 1250 löscht also eine Bildschirmzeile und schreibt dann an deren Anfang die Worte »NEUER INHALT:«.
1270 und 1310	setzen die Flags für Änderungen an der Datei auf 99. Diese Flags werden bei einem Speichervorgang wieder auf 0 gesetzt.
5000	wird nur angesprungen, wenn man versucht, das Programm zu beenden oder ein neues File einzulesen, bevor Änderungen gespeichert wurden.
5030	gibt die Möglichkeit, die Diskette zu wechseln oder auch den Speichervorgang abzubrechen.
5070	fragt nach dem neuen Dateinamen, gibt gleichzeitig den ursprünglichen Namen als »default« ins Eingabefeld.
5100-5140	überprüft, ob auf der eingelegten Diskette ein File mit gleichem Namen vorkommt, der dann in Zeile
5300-5380	sicherheitshalber in den Namen »UTHELP« umbenannt wird, also nicht sofort verloren geht.
5400-5480	enthält den eigentlichen Speichervorgang. Gespeichert wird unter Gebrauch der Variablen N2$ (gefüllt in Zeile 5070).
9700	Bei Eingabe eines »E« anstelle einer laufenden Nummer (Zeile 1000) wird das Programm beendet, wenn eventuell Änderungen gespeichert wurden oder die Frage nach Speichern verneint wurde.

Tabelle 2. Programmstruktur von »P++Z«

nennung ist auf dem Bildschirm angezeigt. Sollten Sie nachträglich bemerken, daß beim Verändern einer Datei Fehler entstanden sind, so können Sie unter »UTHELP« auf die alte Datei zurückgreifen. Da auf jeder Diskette nur eine Datei mit dem Namen »UTHELP« gespeichert werden kann, sollten Sie vor dem Bearbeiten einer weiteren Datei Ihre letzten, mit »P++Z« vorgenommenen Änderungen überprüfen.

Vor dem Laden einer anderen Datei kontrolliert das Programm, ob ausgeführte Änderungen gespeichert wurden. Haben Sie dies noch nicht getan, so fragt Sie der Computer, ob Sie eine Speicherung wünschen.

Als weitere Programmierhilfen bieten wir Ihnen für »P++Z« eine Übersicht zur Programmstruktur (Tabelle 2) und die Variablenliste mit den Zeilennummern, in denen die Variable vorkommt (Tabelle 1).

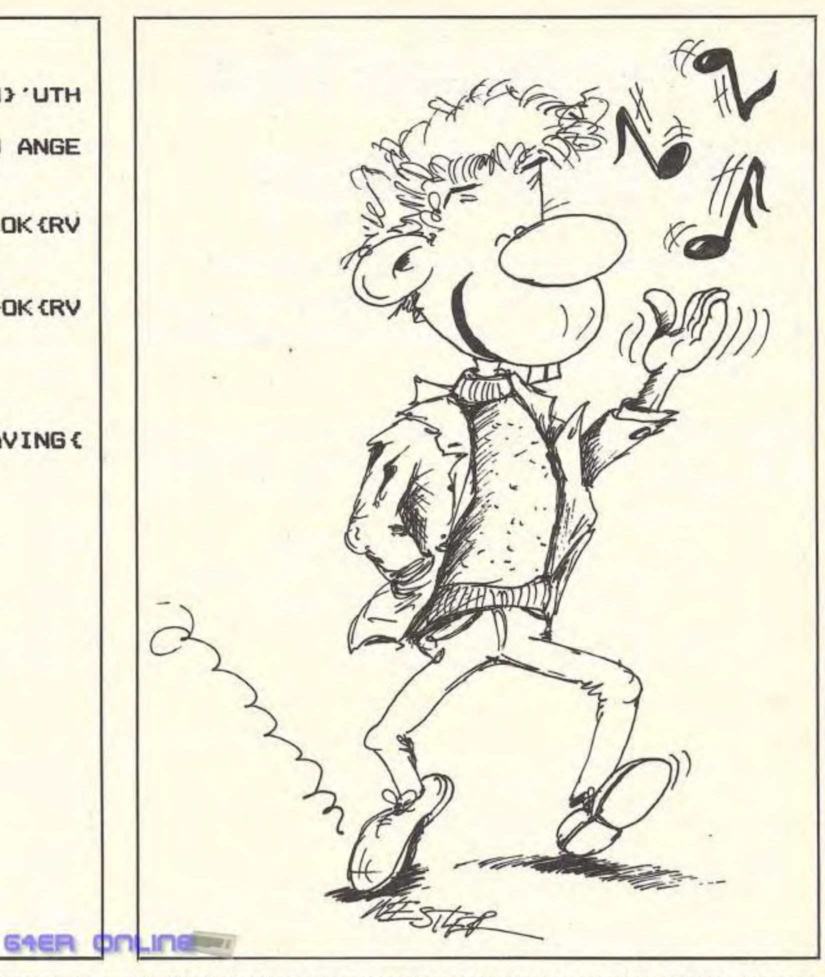
(Herbert Wißmann/kn)

```
1 REM"@P++Z":REM  6.10.85  PSZ=2586
3 REM  **************
4 REM  * H.WISSMANN *
5 REM  * 02822 4922 *
6 REM  *  30.11.83  *
7 REM  **************
19 GOTO190
24 CLOSE15:OPEN15,8,15
25 INPUT#15,EN,EM$,ET,ES:IFEN<2THENRETUR
N
27 IFEN=64THENQ$="{DOWN,2SPACE}IST KEIN
SEQ.FILE{SPACE,FLASH ON}!{FLASH OFF,DOWN
}":RETURN
28 IFEN=62THENQ$="{DOWN,2SPACE}GIBT ES N
ICHT{SPACE,FLASH ON}!{FLASH OFF,DOWN}":R
ETURN
29 PRINT"{DOWN }"EM$"{SPACE,RVSON,FLASH
ON}ERROR{FLASH OFF,RVOFF}":CLOSE15:END
190 GOSUB24
193 S2$="********************************
*********"
194 S1$="********************************
*********"
195 GF$=CHR$(34)
200 PRINT"{CLR,BLUE}**{6SPACE}SEQ.FILES
LESEN/AENDERN{7SPACE}**{2DOWN,2SPACE}FIL
E NAME{DOWN}"
210 N1$="":INPUTN1$:IFN1$=""THEN9720
220 N3$="{RVSON}"+N1$+"{RVOFF}"
230 PRINT"{DOWN,2SPACE}BESTAND ";N3$
240 :
250 REM ** REC.ZAEHLEN
260 OPEN2,8,3,N1$+",S,R"
270 Q$="":GOSUB25:IFQ$<>""THENCLOSE2:PRI
NTQ$:GETKEYQ$:GOTO200
280 I=1
290 INPUT#2,Q$
300 IFST=64THEN330
310 IFST<>0THEN25
320 I=I+1:GOTO290
330 CLOSE2
340 GS=I:DIMA$(I)
350 IFGS>30THEN PRINT GS;" VARIABLE, EIN
EN MOMENT BITTE{DOWN}"
360 :
370 REM **   REC.LESEN
380 OPEN2,8,3,N1$+",S,R"
390 Q$="":GOSUB25:IFQ$<>""THENCLOSE2:PRI
NTQ$:GETKEYQ$:GOTO200
400 I=1
410 INPUT#2,A$(I)
420 IFST=64THEN450
430 IFST<>0THEN25
440 I=I+1:GOTO410
450 CLOSE2
460 :
1000 PRINT"{CLR,6SPACE}DATEIINFO ";N3$;"
 ";GS;"VARIABLE{DOWN}"
1010 PRINTS1$" BITTE GEWUENSCHTE LFD.NUM
MER EINGEBEN"
1020 PRINT" ODER 0=NEUER FILE, C=COPY, E
=ENDE"
1030 PRINT"{6SPACE}A=ALLE VARIABLEN LIST
EN":PRINTS1$"{DOWN}"
1040 Q$="":INPUT"{UP }WELCHE NR: ";Q$
1050 IFQ$="A"THEN1150
1060 IFQ$="0"ORQ$="E"THEN9700
1070 IFQ$="C"THEN5030
1080 Q%=VAL(Q$):IFQ%<1ORQ%>GSTHEN1040
1090 PRINT"{2DOWN}";S2$;"{DOWN }";GF$;A$
(Q%);GF$:PRINT"{DOWN}";S2$;"{DOWN }A=AEN
DERN{2SPACE}L=LOESCHEN"
1100 GETKEYQ$:IFQ$="A"THEN1240
1110 IFQ$="L"THEN1290
1120 GOTO1000
1130 :
1140 REM ** DATEI LISTEN
1150 PRINT"{CLR,3SPACE}"N3$"{DOWN}"
1160 FORL=1TOGS
1170 IFPEEK(205)<23THEN1190
1180 PRINTN3$;"WEITER NACH RETURN";:GETK
EYQ$:PRINT"{CLR}";N3$:IFQ$="E"THEN1000
1190 PRINTUSING"### ";L;
1200 PRINT"= ";GF$;A$(L);GF$;"{DOWN}"
1210 NEXT
1220 GETKEYQ$:GOTO1000
1230 :
1240 REM ** AENDERN  A$(Q%)
1250 PRINT"{UP}"CHR$(27)"QNEUER INHALT:"
1260 INPUTA$(Q%)
1270 KT=99
1280 GOTO1000
1290 REM ** LOESCHEN
1300 A$(Q%)=""
1310 KT=99
1320 GOTO1000
1330 :
5000 PRINT"{CLR,DOWN }> AENDERUNG ABSPEI
CHERN{2DOWN}":INPUT" JA/NEIN";X$
5010 IFX$="NEIN"THEN9710:ELSE IFX$<>"JA"
THEN5000
5030 PRINT"{CLR,2SPACE}DATEI SPEICHERN"
5040 PRINT"{2DOWN }BITTE GEWUENSCHTE ZIE
LDISKETTE EINLEGEN UND{SPACE,RVSON}RETUR
N{RVOFF,2SPACE}(Z=ZURUECK)"
5050 GETKEYQ$:IFQ$="Z"THEN1000
5060 PRINT"{DOWN }UNTER WELCHEM NAMEN SO
LL DIE DATEI{6SPACE}GESPEICHERT WERDEN{S
PACE,DOWN}"
5070 PRINT"{2SPACE}"N1$:INPUT"{UP}";N2$:
IFN2$=""THEN1000
5080 IFLEN(N2$)>15THENN2$=LEFT$(N2$,15)
5090 PRINT"{UP,2SPACE,RVSON}";N2$;"{RVOF
F,2DOWN}"
5100 OPEN5,8,2,N2$+",S,R"
5120 INPUT#15,EN,EM$,ET,ES
5130 CLOSE5:IFEN=62THEN5400
5140 IFEN<>0THEN29
5300 PRINT"{CLR,DOWN }DIE ALTE DATEI "N3
```

Listing zu »P++Z«.
Bitte beachten Sie die Eingabehinweise auf Seite 76.

```
$:PRINT"{DOWN }WIRD NUN{SPACE,RVSON}'UTH
ELP'{RVOFF}.{DOWN}"
5310 PRINT" DANACH WIRD '"N2$"' NEU ANGE
LEGT."
5320 PRINT#15,"S:UTHELP"
5330 GOSUB24:PRINT"{2SPACE,RVSON}S-OK{RV
OFF}"
5350 PRINT#15,"R:UTHELP=";N1$
5360 GOSUB24:PRINT"{2SPACE,RVSON}R-OK{RV
OFF}"
5380 :
5400 KT=0
5410 OPEN1,8,2,N2$+",S,W"
5420 GOSUB25:PRINT"{2SPACE,RVSON}SAVING{
RVOFF }"N2$
5430 FORL=1TOGS
5440 IFA$(L)=""THEN5470
5450 PRINT#1,A$(L)
5460 GOSUB25
5470 NEXT
5480 CLOSE1:GOTO1000
5620 :
9700 IFKT=99THENGOTO5000
9710 IFQ$="0"THENCLR:GOTO190
9720 GOSUB25:CLOSE15
9730 FORL=1TO30:PRINT:NEXT
9740 PRINT"ENDE"
9750 END
```

Listing zu »P++Z« (Schluß)



Kurvenplotten mit Hardcopy

Mit Hilfe dieses Programmes ist es auf dem C 16 möglich, eine oder mehrere Funktionen in ein Koordinatensystem zu plotten und anschließend eine Hardcopy herzustellen.

Da der C 16 bei hochauflösender Grafik leider nur noch 2 KByte für Basic übrig hat, mußte auf REM-Zeilen verzichtet werden, wodurch die Übersichtlichkeit des Programms etwas leidet.

Das Programm gliedert sich in folgende Teile:

10-190 Initialisierung, Eingabe der Funktion, Wahl von Koordinatensystem und Maßstab
200-320 Funktion wird geplottet
330-390 Festlegung des Nullpunktes
400-480 Festlegung der Vergrößerung
510-610 Koordinatensystem und Beschriftung zeichnen
620-710 Bildschirm lesen und Ausgabe an Drucker

Nach dem Starten des Programmes mit RUN muß zunächst die gewünschte Funktion eingegeben werden. Dies geschieht dadurch, daß in der aufgelisteten Zeile 200 die gewünschte Funktion anstelle von F(x) eingegeben wird (x muß als Variable benutzt werden).

Durch Drücken von RETURN wird die Zeile programmiert. Wenn Sie wünschen, daß der hochauflösende Bildschirm gelöscht werden soll, so drücken Sie nun F1; soll die neue Funktion dagegen zu bestehenden Kurven hinzugeplottet werden, so drücken Sie F3.

Nun müssen Sie den Maßstab eingeben (für eine Sinusfunktion ist beispielsweise 40 ein guter Wert) und eines von den drei möglichen Koordinatensystemen auswählen.

Bei der Eingabe der Funktion braucht man nicht auf Definitionslücken zu achten, da die TRAP-Anweisung in Zeile 240 »DIVISION BY ZERO« Fehler und andere abfängt.

Nach Beendigung des Plottens erscheint in der rechten oberen Ecke ein »E«. Drückt man jetzt auf »C«, wird der Bildschirm gelesen und als Hardcopy über den Drucker ausgegeben (siehe Bild). Dies dauert leider fast 30 Minuten, da

```
10 REM CHRISTIAN SCHOSKE
20 KEY1,"O=1:GOTO90"+CHR$(13)
30 KEY 2,"RUN"+CHR$(13)
40 KEY 3,"O=0:GOTO90"+CHR$(13)
50 GRAPHIC0
60 PRINT"{CLR}GEBEN SIE FUER F(X) DIE FU
NKTION EIN.{3SPACE}ANSCHLIESSEND{SPACE,R
VSON}RETURN{RVOFF }DRUECKEN"
70 PRINT"{RVSON}F1{RVOFF }BILDSCHIRM LOE
SCHEN":PRINT"{RVSON}F3{RVOFF }BILDSCHIEM
 NICHT LOESCHEN":PRINT
80 PRINT"200 DEF FNA(Q)=F(X)" :END
90 REM
100 GRAPHIC0:PRINT:PRINT:INPUT"MASSTAB "
;SC
110 PRINT:PRINT"KOORDINATENSYSTEM "
120 PRINT:PRINTSPC(3)"N"SPC(8)"N"SPC(9)"
N"SPC(17)"1{2SPACE}N"SPC(8)"N{2SPACE}2"S
PC(6)"N{2SPACE}3"
130 PRINT"PPP£PPPP"SPC(4)"N"SPC(9)"NPPPP
PPP"SPC(13)"N"SPC(8)"N"SPC(9)"N "
140 PRINTSPC(3)"N"SPC(8)"N"SPC(9)"N"SPC(
20)"N"SPC(8)"NPPPPPPPP N"
150 GETKEYA$:A%=VAL(A$):IFA%<1ORA%>3THEN
150
160 COLOR1,1,1
170 GRAPHIC1,O:CHAR1,38,0," "
180 ONA%GOTO 330,350,370
190 REM
200 DEF FNA(Q)=SQR(X)
210 Z=0
220 DO
230 X=(-AX+Z)/SC
240 TRAP270
250 Y=SC*(FNA(X))
260 DRAW1,Z,AY-Y
270 Z=Z+1
280 LOOPWHILEZ<320
290 CHAR1,38,0,"E"
300 GETKEYA$:IFA$="C"THEN620:ELSEGRAPHIC
0
310 PRINT"{CLR}F1 NEUER MASSTAB":PRINT"F
2 NEUE FUNTION{UP}"
320 END
330 AY=100:AX=160:DRAW1,0,AYTO320,AY:DRA
W1,AX,0TOAX,200
340 GOTO380
350 AY=199:AX=0:DRAW1,AX,0TOAX,AY:DRAW1,
AX,AYTO320,AY
360 GOTO380
370 AY=100:AX=0:DRAW1,AX,0TOAX,200:DRAW1
,AX,AYTO320,AY
380 GOSUB400:IFO<>0THENGOSUB510
390 GOTO190
400 WE =320/SC
410 IFWE >=230THENM=50
420 IFWE >=200ANDWE <230THENM=25
430 IFWE >=120ANDWE <200THENM=20
440 IFWE >= 65ANDWE <120THENM=10
450 IFWE >= 26ANDWE < 65THENM=5
460 IFWE >= 13ANDWE < 26THENM=2
470 IFWE<13THENM=1:F=1
480 RETURN
510 YA=AY:    XA=AX:MC=0:IFM>1THENF=1
520 DO
530 FORD=1TOM
540 XA=XA+SC:YA=YA-SC:IF(XA/F)>312THENEX
IT
550 NEXTD:MC=MC+1
560 IF(XA/F/8)-1>39THEN580
570 DRAW1,(XA/F),AYTO(XA/F),(AY-5):CHAR1
,(XA/F/8-1),(AY/8)-1,STR$(MC*M)
580 IF(YA/F/8)<1THEN600
590 DRAW1,AX,(YA/F)TO(AX+5),(YA/F):CHAR1
,(AX/8+1),(YA/F/8),STR$(MC*M)
600 LOOP
610 RETURN
620 OPEN4,4:Z=0:CMD4
630 FORA=0TO27
640 FORB=0TO31
650 FORC=0TO10
660 FORD=0TO6:LOCATE(B*10+C),(A*7+D):IFR
DOT(2)=1THENZ=Z+2↑D
670 NEXTD
680 Z=Z+128:A$=A$+CHR$(Z):Z=0:NEXTC
690 PRINTCHR$(8)A$;:A$="":NEXTB
700 PRINT" ":NEXTA
710 PRINT#4,CHR$(15):CLOSE4:GOTO300
```

©64'er

Listing »Funktionenplotter C 16«

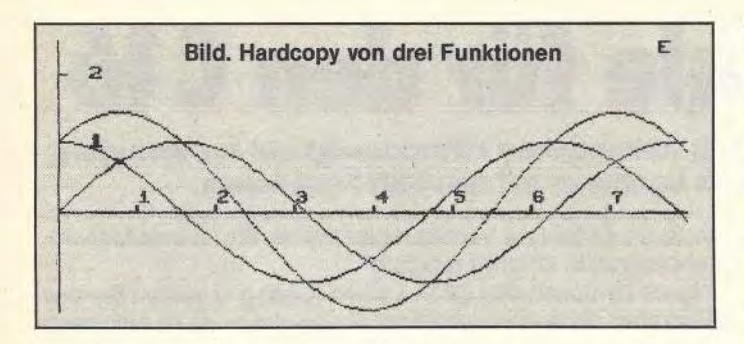


Bild. Hardcopy von drei Funktionen

jeder der 64000 Bildpunkte einzeln ausgelesen und in ein dem Drucker genehmes Bitmuster umgewandelt werden muß. Dieser Programmteil (ab Zeile 620) wurde für den Seikosha GP100VC geschrieben und muß für andere Drucker eventuell geändert werden. Jeder andere Tastendruck führt zu einem kurzen Menü.

Man kann durch Druck auf F1 Maßstab und Koordinatensystem neu wählen.

Durch Druck auf F2 kommt man wieder zur Anfangsroutine und kann eine neue Funktion eingeben.

(Christian Schoske/ev)

HELP und TRACE verbessert

Dieses kleine Programm für den C16 implementiert eine wesentlich erweiterte TRACE-Funktion. Auch HELP wird damit um einiges übersichtlicher. Statt wildem Blinken eine augenschonende Anzeige!

Nachteilig bei der bisherigen Fehlersuche durch »HELP« ist das entnervende Blinken der Fehlerstelle und aller nachfolgenden Zeichen. Eine konzentrierte Fehlersuche ist kaum möglich. Durch die Änderung dieser Routine wird die Fehlerstelle fortan nicht mehr blinkend, sondern in reverser Schreibweise ausgegeben und auch nur diese einzige Stelle, nicht mehr die gesamte Restzeile. Der Fehler ist somit mit einem einzigen Blick zu erfassen.

Das C16-Tracing mag etwas für Leute mit Facettenaugen sein, aber wohl nichts für die Mehrzahl der Anwender. In der durch dieses Programm erzeugten neuen Version werden nicht mehr wild die abgearbeiteten Zeilennummern ausgegeben, sondern die gesamte aktuelle Programmzeile gezeigt. Der Befehl, auf dem der Programmzeiger gerade steht, erscheint revers geschrieben. Nach jeder Zeilenausgabe stoppt das abzuarbeitende Programm, und die Trace-Routine wartet auf irgendeinen Tastendruck.

Danach wird die angezeigte Zeile bis zum folgenden Trennzeichen ausgeführt. Dieser Vorgang wiederholt sich bis zum Programmende. Der Programmablauf läßt sich wie gewohnt mit der STOP-Taste unterbrechen.

Vor allem für Basic-Anfänger ist dieses verbesserte Trace eine wertvolle Hilfe, da sich das Programm bei der Ausführung von Befehl zu Befehl direkt verfolgen läßt.

(Wolfgang W. Wirth/ev)

```
180 REM ****************************
190 REM
200 REM
220 ADRESSE=819:ANZAHL=8
230 FOR ZEILE=410 TO 760 STEP 10
240 SUMME=0
250 IF ZEILE=760 THEN ANZAHL=3
260 FOR SPALTE=1 TO ANZAHL
270 READ BYTE$:BYTE=DEC(BYTE$)
280 SUMME=SUMME+BYTE AND 255
290 POKE ADRESSE,BYTE
300 ADRESSE=ADRESSE+1
310 NEXT
320 READ TEST$
330 IF SUMME=DEC(TEST$) THEN 360
340 PRINT"FEHLER IN ZEILE";ZEILE
350 FLAG=1
360 NEXT
370 IF FLAG THEN END
380 SYS 1082
400 REM
410 DATA38,66,53,A0,03,84,49,84,E5
420 DATA0F,20,5F,A4,A9,20,A4,49,E8
430 DATA29,7F,20,B2,90,C9,22,D0,C5
440 DATA06,A5,0F,49,FF,85,0F,C8,5E
450 DATAF0,09,A2,00,86,C2,24,53,5A
460 DATA10,19,A6,60,98,18,65,5F,A3
470 DATA90,01,E8,EC,F6,04,D0,0B,3A
480 DATACD,F5,04,90,06,F0,04,26,76
490 DATAC2,46,53,20,D1,04,F0,E3,23
500 DATA10,C8,C9,FF,F0,C4,24,0F,87
510 DATA30,C0,AA,84,49,A0,81,84,0C
520 DATA23,A0,8E,84,22,A0,00,CA,61
530 DATA10,0F,B1,22,48,E6,22,D0,12
540 DATA02,E6,23,68,10,F4,30,EF,96
550 DATAC8,B1,22,30,99,20,B2,90,C6
560 DATAD0,F6,60,AE,EF,04,E8,F0,9F
570 DATA19,AD,F0,04,AC,F1,04,85,E0
580 DATA14,84,15,20,3D,8A,90,0A,2E
590 DATA20,3E,90,A6,14,A5,15,20,82
600 DATA33,03,4C,3E,90,20,73,04,E7
610 DATA20,D9,03,4C,DC,8B,F0,83,22
620 DATA2C,EB,02,10,2E,24,81,10,0C
630 DATA2A,48,A4,3C,A6,3B,D0,01,04
640 DATA88,CA,8E,F5,04,8C,F6,04,5F
650 DATAA5,39,A4,3A,85,14,84,15,EE
660 DATA20,3D,8A,A6,14,A5,15,20,7B
670 DATA33,03,20,3E,90,20,DD,EB,0C
680 DATAF0,FB,68,C9,EA,F0,03,4C;45
690 DATA3F,8C,20,73,04,4C,AE,03,5F
700 DATAA2,FF,86,3A,20,5A,88,86,E9
710 DATA3B,84,3C,20,73,04,AA,F0,2C
720 DATAEF,90,09,20,53,89,20,79,1D
730 DATA04,4C,D3,03,4C,2E,87,A2,C9
740 DATA03,8E,09,03,E8,8E,03,03,19
750 DATAA2,D0,8E,08,03,A2,1B,8E,56
760 DATA02,03,60,65
```

©64'er

Listing »Extended Help & Trace« für den C 16

»Fenster«-Befehle für den C16

Dieses Programm stellt den im C 16-Basic leider nicht vorhandenen »Window«-Befehl zur Verfügung und macht Schluß mit dem umständlichen Hantieren mit den ESC-Funktionen.

Wer im Befehlssatz des Basic 3.5 nach dem Befehl »Window« sucht, wird nicht fündig. Die groß angekündigte moderne Window-Technik läßt doch zu wünschen übrig. Ein Window, also ein Bildschirmfenster, kann nur mit Hilfe der ESC-Taste definiert werden.

So muß man, um ein Window zu setzen, mit dem Cursor an die linke obere Ecke des gewünschten Bildschirmfensters fahren und die Tasten ESC und T drücken. Die rechte untere Ecke des Fensters wird ebenfalls mit dem Cursor gewählt und mit ESC und B gesetzt. Erst jetzt ist das Fenster definiert.

Im Direktmodus ist dieser Aufwand nicht hinderlich, sondern bringt eher Vorteile, da kein Befehl »WINDOW« am Bildschirm erscheint. Im laufenden Programm ist es jedoch sehr umständlich, ein Fenster zu generieren. Es sind sehr viele PRINT-Anweisungen nötig.

Da die Window-Funktion jedoch in vielen Programmen nützlich ist, lohnt es sich schon, diese Funktion komfortabel als Basic-Befehl zur Verfügung zu haben. Ein kleines Maschinenprogramm macht's möglich.

Nach Eintippen des DATA-Laders (Listing 1) sollten Sie das Programm als erstes unbedingt abspeichern, da es sich nach Kontrolle der Prüfsumme selbst initialisiert und anschließend der DATA-Lader gelöscht wird.

Haben Sie das Programm abgespeichert, dann können Sie es mit »RUN« starten und haben kurz danach den neuen Basic-Befehl »FENSTER« zur Verfügung.

Der FENSTER-Befehl hat folgendes Format und kann in jedem Basic-Programm verwendet werden:
FENSTER AZ,AS,EZ,ES

Die Parameter haben folgende Bedeutung:

AZ = AnfangsZeile
AS = AnfangsSpalte
EZ = EndZeile
ES = EndSpalte

```
10 REM **********************************
20 REM *                                *
30 REM *    FENSTER FUER C 16 & 116     *
40 REM *                                *
50 REM *    CHRISTIAN QUIRIN SPITZNER   *
60 REM *    GRUBERSTRASSE 53            *
70 REM *    8011 POING                  *
80 REM *                                *
90 REM **********************************
100 :
110 :
120 REM *** SPEICHER BEGRENZEN ***
130 :
140 POKE 53,255 : POKE 54,62 : POKE 55,2
55 : POKE 56,62 : CLR
150 :
160 REM *** ERKLAERUNG ***
170 :
180 PRINT"{CLR,DOWN}"TAB(12)"***{SPACE,F
LASHON,RED}FENSTER{FLASHOFF,BLACK }***"
190 PRINT"{2DOWN,6SPACE}FORMAT:"
200 PRINT"{DOWN,6SPACE,PURPLE,RVSON}FENS
TER AZ,AS,EZ,ES{BLACK}"
210 PRINT"{DOWN,6SPACE}AZ = ANFANGSZEILE
"
220 PRINT"{6SPACE}AS = ANFANGSSPALTE"
230 PRINT"{6SPACE}EZ = ENDZEILE"
240 PRINT"{6SPACE}ES = ENDSPALTE"
250 PRINT"{DOWN,6SPACE}NICHT IM DIREKTMO
DUS !"
260 :
270 REM DATA'S EINLESEN ***
280 :
290 FOR I=16128 TO 16383
300 :  READ P
310 :  Z=Z+P
320 :  POKE I,P
330 NEXT I
340 IF Z <> 30097 THEN PRINT"{DOWN,6SPAC
E,FLASHON}DATA ERROR{FLASHOFF}":END
350 :
360 REM *** INITIALISIEREN ***
370 :
380 SYS 16128
390 :
400 REM *** FENSTER-BEFEHL AUFRUFEN ***
410 :
420 FENSTER 4,1,25,40
430 :
440 REM *** BASIC-LOADER LOESCHEN ***
450 :
460 NEW
470 :
480 REM *** DATA'S ***
490 :
500 DATA 169,024,141,008,003,169,063,141
510 DATA 009,003,162,000,189,205,063,032
520 DATA 210,255,232,224,043,208,245,096
530 DATA 162,000,032,115,004,221,209,063
540 DATA 208,007,232,224,007,208,243,240
550 DATA 006,032,121,004,076,217,139,032
560 DATA 183,251,032,115,004,032,132,157
570 DATA 142,249,063,032,145,148,032,132
580 DATA 157,142,250,063,032,145,148,032
590 DATA 132,157,142,251,063,032,145,148
600 DATA 032,132,157,142,252,063,174,249
610 DATA 063,224,000,208,003,076,161,148
620 DATA 224,024,016,249,174,251,063,224
630 DATA 026,016,242,236,249,063,048,237
640 DATA 174,250,063,224,000,240,230,224
650 DATA 041,016,226,172,252,063,224,041
660 DATA 016,219,236,250,063,048,214,162
670 DATA 000,222,249,063,232,224,004,208
680 DATA 248,173,249,063,133,205,173,250
690 DATA 063,133,202,169,027,032,210,255
700 DATA 169,084,032,210,255,173,251,063
710 DATA 133,205,173,252,063,133,202,169
720 DATA 027,032,210,255,169,066,032,210
730 DATA 255,169,019,032,210,255,032,193
740 DATA 251,032,121,004,076,217,139,000
750 DATA 000,000,000,000,000,147,013,032
760 DATA 130,070,069,078,083,084,069,082
770 DATA 132,032,040,067,041,032,067,072
780 DATA 082,073,083,084,073,065,078,032
790 DATA 081,085,073,082,073,078,032,083
800 DATA 080,073,084,090,078,069,082,013
810 DATA 013,000,000,000,000,000,000,000
```

©64'er

Listing 1. Der DATA-Lader zum »Fenster«-Befehl

Gibt man im Programm beispielsweise die folgende Zeile 10 ein, so wird ein Fenster von Spalte 5 der fünften Zeile bis einschließlich Spalte 35 der 15. Zeile definiert.

```
10 FENSTER 5,5,35,15
```

Das Fenster kann durch zweimaliges Drücken der HOME-Taste oder durch Generieren eines neuen Fensters gelöscht werden.

Dem Fenster-Befehl müssen auf jeden Fall vier Parameter folgen. Für die Parameter AZ und EZ sind nur Zahlen zwischen 1 und 25 zulässig. AS und ES dürfen nur zwischen 1 und 40 liegen. Außerdem ist zu beachten, daß AZ auf keinen Fall größer als EZ, und AS nicht größer als ES gewählt wird.

Der Fenster-Befehl darf auch nicht im Direktmodus verwendet werden. Sollten diese Bedingungen nicht beachtet werden, erscheint die Fehlermeldung »SYNTAX ERROR«.

Das Beispielprogramm Fenster-Demo (Listing 2) generiert zufällige Textfenster und füllt diese farbig aus.

Der FENSTER-Befehl verkürzt ein Basic-Programm bei häufiger Anwendung gegenüber der konventionellen Methode über PRINT-Anweisungen ganz erheblich, so daß die vom Maschinenprogramm beanspruchten 255 Byte trotz des mageren C 16-Speichers kaum ins Gewicht fallen dürften.

(Christian Quirin Spitzner/ev)

```
100 REM FENSTER-DEMO
110 :
120 COLOR 0,1 : COLOR 4,1 : SCNCLR
130 :
140 DO
150 : AZ = INT(RND(TI)*12)+1
160 : EZ = AZ*2+1
170 : AS = INT(RND(TI)*20)+1
180 : ES = AS*2
190 :
200 : FENSTER AZ,AS,EZ,ES
210 :
220 : PRINT"{CLR}";
230 : COLOR 1,RND(TI)*15+2,RND(TI)*8
240 : E = (ES-AS+1)*(EZ-AZ+1)-1
250 : FOR I=1 TO E
260 :    PRINT"{RVSON,SPACE,RVOFF}";
270 : NEXT
280 : PRINT"{LEFT,RVSON}";CHR$(148);"{SP
ACE,RVOFF}";
290 LOOP
```

©64'er

Listing 2. Ein Demo-Programm zum »Fenster«-Befehl

Zeichensatz selbstgemacht

Wollen Sie einen eigenen Zeichensatz definieren? »Charmachine« ermöglicht Ihnen dies in einer komfortablen Weise. Da Sie auch mehrfarbige Zeichen erzeugen können, ist es besonders für Grafiken interessant.

Die Möglichkeiten des C 16/116 im Bereich der hochauflösenden Bitmap-Grafik sind schier riesig. Doch für viele Programme, die grafischer Unterstützung bedürfen, sind die nach dem Befehl GRAPHIC 1-4 noch vorhandenen 2 KByte nicht ausreichend. In diesem Fall muß der ebenfalls hochauflösende und/oder mehrfarbige Zeichensatz aushelfen. »Charmachine« dient dazu, diesen bequem editieren zu können.

Nun zum Umgang mit dem Programm. Nachdem Sie das Programm (Listing) eingetippt und gestartet haben, erscheint in der linken oberen Ecke des Bildschirms ein 8x8 Punkte großes Feld. Rechts daneben sind die Befehle von »Charmachine« zu sehen. Im unteren Teil des Bildschirms befindet sich ein Feld, in dem alle 256 möglichen Zeichen abgebildet sind. Es handelt sich dabei um die positiven Zeichen beider Zeichensätze (also des Groß-Klein- und des Groß-Grafik-Satzes). Die negativen Zeichen beim C 16/116 können nicht gesondert verändert werden, sie zeigen immer genau die Negative ihrer entsprechenden positiven Zeichen.

In diesem Feld können Sie einen Cursor bewegen und sich das zu verändernde Zeichen aussuchen. Das Zeichen unter dem Cursor wird unterhalb des 8x8-Feldes hochauflösend (hiRes) und mehrfarbig (multicolor) dargestellt. Steht nun der Cursor auf dem richtigen Zeichen, so drücken Sie die Taste P (P=pick). Das Zeichen wird augenblicklich im 8x8-Punktefeld dargestellt, und der Cursor befindet sich ebenfalls dort.

Sie können nun das Zeichen ändern mit SPACE für Punkt löschen (SPACE = unplot), Z für Punkt setzen (Cursor nach rechts) und X für Punkt setzen (Cursor nach unten) (Z/X = plot). Man kann das ganze Feld löschen mit C (=Clear) und invertieren mit R (=reverse). Ist das Zeichen fertig editiert, so kehren Sie mit M (=select mode) in den Wählmodus zurück. Hier können Sie mit den Tasten 1 und 2 (=colors) die Multicolorfarben des Zeichens über »multi« ändern sowie den Zeichensatz speichern mit S (= SAVE) und neu laden mit L (=LOAD von Diskette). Der Befehl C (=COPY) erlaubt Ihnen, das sich noch im 8x8-Feld befindliche Zeichen in ein anderes Zeichen zu kopieren, das mit dem Cursor ausgesucht wird.

Wenn Sie einen fertigen gespeicherten Zeichensatz in ein eigenes Programm einbauen wollen, so müssen Sie diesen mit LOAD "name" 8,1 laden und dann den Befehl NEW eingeben. An den Anfang Ihres Programms müssen Sie die Befehle setzen, die in Zeile 10 von »Charmachine« (Listing) stehen. Sie dienen dazu, den Zeichensatz vor dem Überschreiben durch Strings zu schützen. Der C 16/116 hat dann noch 10 KByte Speicher frei. Um den Zeichensatz schließlich einzuschalten, müssen Sie folgende POKEs eingeben:

POKE 65298,0 (Flag für Zeichensatz aus ROM oder RAM),
POKE 65299,56 (Anfangsadresse des Zeichensatzes im RAM),
POKE 65287,8+(16)+128

Beim letzten Befehl steht 8 für 40 Zeichen, (16) für Multicolor (wahlweise), 128 für Darstellung der Zeichensätze Groß+Klein und Groß+Grafik. Ohne 128 wird nur einer dieser Zeichensätze, aber mit seinen negativen Zeichen, dargestellt.

Der undefinierte Zeichensatz liegt im Speicherbereich von 14336 bis 16383 ($3800 bis $3FFF).

Nun zu einigen Besonderheiten des Programms. Beim Kopieren des Zeichensatzes vom ROM ins RAM greift das Programm auf den Monitor des C 16/116 zurück. Vom Programm aus wird in den Monitor gesprungen und dort mit dem Befehl T (=TRANSVERE) der Zeichensatz von ROM ins RAM geschrieben. Um in den Monitor und wieder zurück zu kommen, wird der »programmierte Direktmodus« benutzt, wobei die erforderlichen Befehle auf den Bildschirm gedruckt werden und durch Auffüllen des Tastaturpuffers mit dem ASCII-

Code der RETURN-Taste nach dem Programmabbruch mit END ausgeführt werden. Dieses wird in den Zeilen 20010 bis 20050 bei »Charmachine« vollzogen.

Dasselbe Verfahren wird beim Speichern des Zeichensatzes in den Zeilen 1052 bis 1060 angewandt, nur mit dem Monitorbefehl S (=SAVE).

Die Maschinenroutinen, die ab der Zeile 21010 eingelesen werden, dienen zum schnellen Umrechnen der Bits eines Zeichens in die Byte-Folge des 8x8-Feldes auf dem Bildschirm, sowie dem umgekehrten Vorgang, Zeichen direkt nach den Punkte-Anordnungen im 8x8-Feld umzudefinieren.

Die erste Routine startet ab der Adresse 864 und wird im Programm in Zeile 620 aufgerufen, die zweite ab 950 und wird in Zeile 720 aufgerufen.

In der abgebildeten Tabelle finden Sie eine Variablenliste für »Charmachine«.

(Jens Engel/kn)

Tabelle. Variablenliste zu »Charmachine«

Variablenliste	
L	= Ladeflag
F1,F2,F3	= Multicolorfarben
T,W	= Schleifenvariablen
X	= Cursorposition horizontal
Y	= Cursorposition vertikal
B	= Zeichen unter Cursor
X1	= Letzte X-Position
Y1	= Letzte Y-Position
AD	= Adresse der linken oberen Ecke des Feldes
A$,B$	= gedrückte Taste
CH	= aktuelles Zeichen
C1	= Adresse des aktuellen Zeichens im Zeichensatz
C2,C3	= Hi-, Lowbyte von C1
Y2	= Speicher für Y
NA$	= Filename
Q	= Lesevariable für READ

```
1 REM ********************************
2 REM ***       CHAR-MACHINE       ***
3 REM ***WRITTEN 1985 BY JENS ENGEL***
4 REM ********************************
9 IFL=1THENL=0:GOTO20
10 POKE51,255:POKE52,55:POKE55,255:POKE5
6,55:CLR
12 GOTO20000
20 POKE65298,0:POKE65299,56:POKE65287,15
2
22 IFPEEK(864)<>169THENGOSUB21000
90 F1=2:F2=0:F3=4:COLOR0,15,2:COLOR1,3,7
:COLOR4,1:COLOR2,2,0:COLOR3,2,4
100 PRINT"{RVSON,CLR,8SPACE}££{3SPACE,RV
SON }CHARMACHINE "
102 PRINT"{RVSON,8SPACE}££"
104 PRINT"{RVSON,8SPACE}££{11SPACE}COMMA
NDS:"
106 PRINT"{RVSON,8SPACE}££{2SPACE}££££££
£££££££££££££££££££££"
108 PRINT"{RVSON,8SPACE}££{2SHFT-SPCE}££
 SPACE=UNPLOT £ X/Z=DOT££"
110 PRINT"{RVSON,8SPACE}££{2SPACE}££ CRS
R=MOVE{4SPACE}£ C=CLEAR££"
112 PRINT"{RVSON,8SPACE}££{SHFT-SPCE }££
 R=REVERSE{4SPACE}£ L=LOAD ££"
114 PRINT"{RVSON,8SPACE}££{2SPACE}££ M=S
ELECT MODE£ S=SAVE ££"
116 PRINT"{RVSON}££££££££££{2SPACE}££ 1,
2 =COLORS{2SPACE}£ P=PICK ££"
118 PRINT"{RVSON,12SPACE}££ N=NEW CHAR{3
SPACE}£ O=COPY ££"
120 PRINT"{RVSON,3SPACE}@{5SPACE}@{2SPAC
E}£££££££££££££££££££££££££££"
122 PRINT" HIRES MULTI"
124 PRINT"{RVSON,DOWN,13SPACE}WRITTEN 19
85 BY JENS ENGEL"
130 FORT=0TO7:FORW=0TO31:POKE3676+W+T*40
,T*32+W:NEXTW,T:POKE3676,86
140 POKE2048+409,92
180 X=0:Y=0:B=0
190 GOTO300
200 X1=7:Y1=7:AD=3072
250 GETKEYA$
255 GOSUB500
258 IFA$="X"THENB=0:A$="{DOWN}":GOSUB500
260 IFA$="Z"THENB=0:A$="{RIGHT}":GOSUB50
0
262 IFA$=" "THENB=32:A$="{RIGHT}":GOSUB5
00
264 IFA$="C"THENPRINT"{HOME}";:FORT=0TO7
:PRINT"{8SPACE}":NEXT:A$="":X=0:Y=0:GOSU
B510
266 IFA$="M"THEN700
268 IFA$="R"THENGOSUB800
290 POKE52,55:GOTO250
300 X1=31:Y1=7:AD=3676
350 GETKEYA$
355 GOSUB500
357 POKE3475,B:POKE3481,B
360 IFA$="P"THENGOTO600
362 IFA$="O"THENCH=Y*32+X:C1=14336+CH*8:
C2=INT(C1/256):C3=C1-C2*256
364 IFA$="O"THENPOKE3,C3:POKE4,C2:SYS950
370 IFA$="1"THENF1=F1+1:IFF1=8THENF1=0:C
OLOR0,15,F1:ELSECOLOR0,15,F1
372 IFA$="2"THENF2=F2+1:IFF2=8THENF2=0:C
OLOR3,2,F2:ELSECOLOR3,2,F2
380 IFA$="S"THEN1000
382 IFA$="L"THEN2000
384 IFA$="N"THENRUN
390 POKE52,55:GOTO350
500 POKEAD+X+Y*40,B
510 IFA$="{RIGHT}"THENIFX<X1THENX=X+1
512 IFA$="{LEFT}"THENIFX>0THENX=X-1
514 IFA$="{UP}"THENIFY>0THENY=Y-1
516 IFA$="{DOWN}"THENIFY<Y1THENY=Y+1
520 B=PEEK(AD+X+Y*40):POKEAD+X+Y*40,86
590 RETURN
600 REM **CHANGE TO EDIT-MODUS
610 CH=Y*32+X:C1=14336+CH*8:C2=INT(C1/25
6):C3=C1-C2*256
615 PRINT"{HOME}";:FORT=0TO7:PRINT"{8SPA
CE}":NEXT
620 POKE3,C3:POKE4,C2:SYS864:B=PEEK(3072
):X=0:Y2=Y:Y=0
640 GOTO200
700 REM **CHANGE TO SELECT-MODUS
710 POKE3072+X+Y*40,B:X=CH-Y2*32:Y=Y2:B=
CH
720 SYS950
740 GOTO300
800 POKE3072+X+Y*40,B:IFB=32THENB=0:GOTO
805
802 B=32
805 FORT=0TO7:FORW=0TO7
810 IFPEEK(3072+T+W*40)=0THENPOKE3072+T+
W*40,32:NEXTW,T:RETURN
820 POKE3072+T+W*40,0:NEXTW,T:RETURN
1000 REM** CHAR SAVEN
1010 PRINT"{CLR,2DOWN}SAVE CHAR-SET:"
1020 PRINT"{DOWN}NAME:";:INPUTNA$
1030 PRINT"{DOWN}OK?":GETKEYB$
1050 POKE65287,8:POKE1339,PEEK(65301)
```

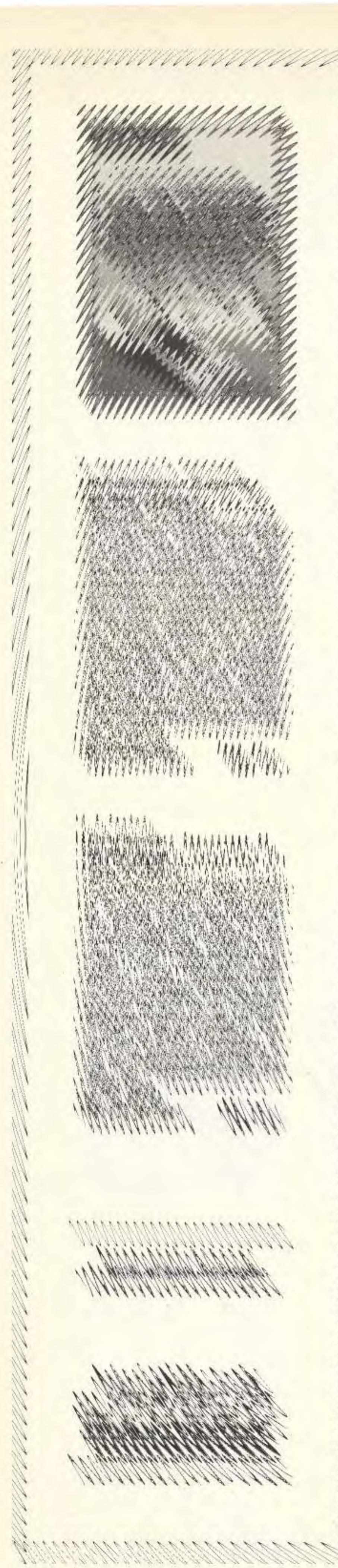
Listing zu »Charmachine«

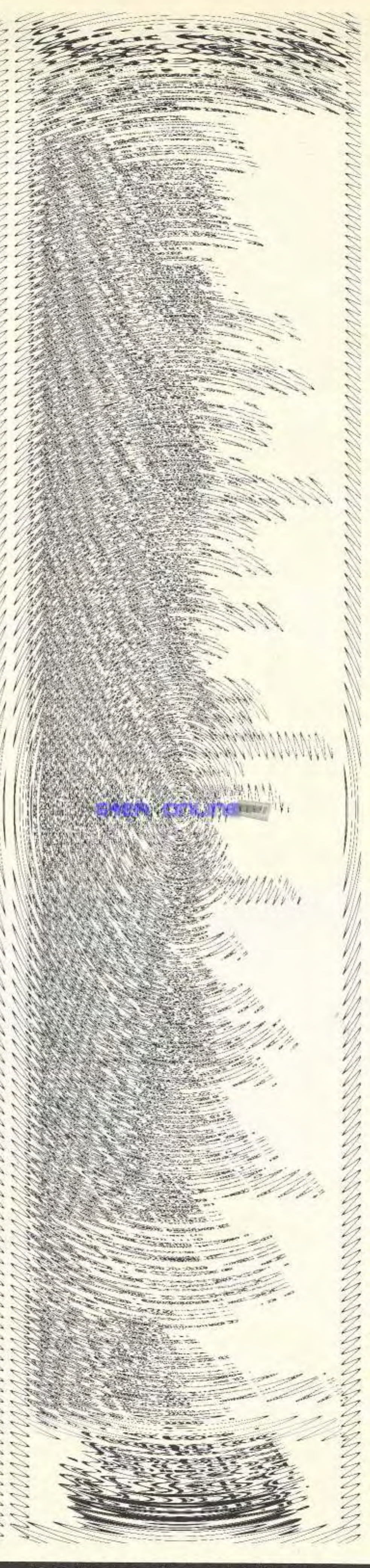
```
1052 PRINT"{CLR}S "CHR$(34);NA$;CHR$(34)
",8,3800,3FFF"
1054 PRINT"{2DOWN}X":PRINT"{DOWN}RUN20":
PRINT"{4DOWN}MO{2UP}";
1060 POKE1319,13:POKE1320,19:FORT=0TO5:P
OKE1321+T,13:NEXT:POKE239,7:END
2000 REM** CHAR LADEN
2010 PRINT"{CLR,2DOWN}LOAD CHAR-SET:"
2020 PRINT"{DOWN}NAME:";:INPUTNA$
2030 PRINT"{DOWN}OK?":GETKEYB$
2040 L=1:LOADNA$,8,1
9999 END
20000 REM **CHAR-SET COPIEREN
20005 POKE1339,PEEK(65301)
20010 PRINT"{CLR,2DOWN}T D000 D800 3800"
20020 PRINT"{DOWN}X"
20030 PRINT"{DOWN}RUN20"
20040 PRINT"{2DOWN}MO{2UP}";
20050 POKE1319,13:POKE1320,19:FORT=0TO5:
POKE1321+T,13:NEXT:POKE239,7:END
21000 REM **MC-DATA EINLESEN
21010 FORT=0TO207:READQ:POKE864+T,Q:NEXT
21020 RETURN
21100 DATA169,0,133,210,133,250,169,12
21102 DATA133,211,160,0,132,5,132,6
21104 DATA162,0,234,234,234,56,102,250
21106 DATA165,250,164,5,49,3,240,6
21108 DATA169,0,160,0,145,210,234,234
21110 DATA230,210,208,2,230,211,232,224
21112 DATA8,208,227,162,0,234,234,234
21114 DATA169,0,133,250,24,165,210,105
21116 DATA32,133,210,165,211,105,0,133
21118 DATA211,234,230,5,165,5,201,8
21120 DATA208,195,96,0,0,0,169,0
21122 DATA133,210,133,250,169,12,133,211
21124 DATA160,0,132,5,132,6,162,0
21126 DATA76,12,4,56,102,250,160,0
21128 DATA177,210,208,10,165,250,164,5
21130 DATA17,3,145,3,234,234,230,210
21132 DATA208,2,230,211,232,224,8,208
21134 DATA227,162,0,234,234,234,169,0
21136 DATA133,250,24,165,210,105,32,133
21138 DATA210,165,211,105,0,133,211,234
21140 DATA230,5,165,5,201,8,208,195
21142 DATA96,0,0,0,169,0,160,0
21144 DATA145,3,200,145,3,200,145,3
21146 DATA200,145,3,200,145,3,200,145
21148 DATA3,200,145,3,200,145,3,76
21150 DATA203,3,0,0,0,0,0,0
```

©64'er

Listing zu »Charmachine« (Schluß).
Bitte beachten Sie die Eingabehinweise auf Seite 76.







Impressum

Herausgeber: Carl-Franz von Quadt, Otmar Weber

Chefredakteur: Michael Scharfenberger

Leitender Redakteur: Albert Absmeier

Koordination: Georg Klinge

Redaktion: Gerd Donaubauer, Volker Everts, Achim Hübner, Gottfried Knechtel, Markus Ohnesorg, Thomas Röder

Titelfoto: Jens Jancke

Layout:
Leo Eder (Ltg.), Sigrid Kowalewski (Cheflayouterin)

Herstellung: Klaus Buck

Auslandsrepräsentation:

Schweiz: Markt & Technik Vertriebs AG, Kollerstr. 3, CH-6300 Zug, Tel. 042-41 56 56, Telex: 862 329

USA: M&T Publishing Inc.; 2464 Embarcadero Way, Palo Alto, CA 94 303

Manuskripteinsendungen: Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten werden, so muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt & Technik Verlag AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Mit der Einsendung von Bauanleitungen gibt der Einsender die Zustimmung zum Abdruck in von Markt & Technik Verlag AG verlegten Publikationen und dazu, daß Markt & Technik Verlag AG Geräte und Bauteile nach der Bauanleitung herstellen läßt und vertreibt oder durch Dritte vertreiben läßt. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

Marketingleiter Vertrieb: Hans Hörl (114)

Vertriebsleitung: Helmut Grünfeldt (189)

Anzeigenverwaltung und Disposition: Michaela Hörl

Verlagsleiter M&T-Buchverlag: Günther Frank

Druck: Druckhaus München, Schellingstr. 39-43, 8000 München 40

Preis: Das Einzelheft kostet DM 14,–

Vertrieb Handelsauflage: Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebs GmbH, Hauptstätter Straße 96, 7000 Stuttgart 1, Telefon (07 11) 6 48 30

Verantwortlich:
Für redaktionellen Teil: Michael Scharfenberger
Für Anzeigen: Britta Fiebig

Redaktions-Direktor: Michael M. Pauly

Vorstand: Carl-Franz von Quadt, Otmar Weber

Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:
Markt & Technik Verlag Aktiengesellschaft, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon (089) 46 13-0, Telex 5-22 052




Bücher zum Commodore 64